References

[1] F. W. Badenhorst and J. Van Tonder, “Determining the factors causing human error deficiencies at a public utility company,” *SA Journal of Human Resource Management*, vol. 2, no. 3, pp. 62–69, 2004.

[2] F. W. Badenhorst and J. Van Tonder, “Determining The Factors Causing Human Error Deficiencies At A Public Utility Company,” *SA Journal of Human Resource Management*, vol. 2, no. 3, 2004, doi: 10.4102/sajhrm.v2i3.47.

[3] J. Rasmussen, A. M. Pejtersen, and L. P. Goodstein, *Cognitive Systems Engineering*. John Wiley & Sons, 1994. [Online]. Available: https://books.google.com/books?id=i2xRAAAAMAAJ&q=10: 0471011983

[4] S. Leatherman and D. M. Berwick, “Accelerating Global Improvements in Health Care Quality,” *JAMA*, vol. 324, no. 24, p. 2479, 2020, doi: 10.1001/jama.2020.17628.

[5] K. Rabieh, A. F. Aydogan, and M. A. Azer, “Towards Safer Roads: An Efficient VANET-based Pedestrian Protection Scheme,” in *2021 International Conference on Communications, Computing, Cybersecurity, and Informatics (CCCI)*, 2021, pp. 1–6. doi: 10.1109/CCCI52664.2021.9583221.

[6] Y. Zhang, L. Jing, Q. Bai, T. Liu, and Y. Feng, “A systems approach to extraordinarily major coal mine accidents in China from 1997 to 2011: an application of the HFACS approach,” *International Journal of Occupational Safety and Ergonomics*, vol. 25, no. 2, pp. 181–193, Apr. 2019, doi: 10.1080/10803548.2017.1415404.

[7] C. Perrow, *Normal Accidents: Living with High Risk Technologies - Updated Edition*. in Princeton paperbacks. Princeton University Press, 2011. [Online]. Available: https://books.google.co.in/books?id=g66J6Vzq6EYC

[8] Y. R. Gunda, S. Gupta, and L. K. Singh, “Assessing human performance and human reliability: a review,” *International Journal of System Assurance Engineering and Management*, vol. 14, no. 3, pp. 817–828, 2023, doi: 10.1007/s13198-023-01893-5.

[9] T. Allahyari *et al.*, “Cognitive Failures, Driving Errors and Driving Accidents,” *International Journal of Occupational Safety and Ergonomics*, vol. 14, no. 2, pp. 149–158, Jan. 2008, doi: 10.1080/10803548.2008.11076759.

[10] R. Kumar, *Research methodology: A step-by-step guide for beginners*. Sage, 2018.

[11] B. Kirwan, *A Guide To Practical Human Reliability Assessment*. CRC Press, 2017. [Online]. Available: https://books.google.co.in/books?id=jwZDDwAAQBAJ

[12] R. Parasuraman and M. Rizzo, *Neuroergonomics: The brain at work*. Oxford University Press, 2008.

[13] C. D. Wickens, "Situation awareness: Review of Mica Endsley's 1995 articles on situation awareness theory and measurement," *Hum Factors*, vol. 50, no. 3, pp. 397–403, 2008.

[14] N. J. Ekanem, A. Mosleh, and S.-H. Shen, "Phoenix – A model-based Human Reliability Analysis methodology: Qualitative Analysis Procedure," *Reliab Eng Syst Saf*, vol. 145, pp. 301–315, 2016, doi: https://doi.org/10.1016/j.ress.2015.07.009.

[15] Y. Li and A. Mosleh, "Dynamic simulation of knowledge based reasoning of nuclear power plant operator in accident conditions: Modeling and simulation foundations," *Saf Sci*, vol. 119, pp. 315–329, 2019, doi: https://doi.org/10.1016/j.ssci.2018.02.031.

[16] J. G. Allen, P. MacNaughton, U. Satish, S. Santanam, J. Vallarino, and J. D. Spengler, "Associations of cognitive function scores with carbon dioxide, ventilation, and volatile organic compound exposures in office workers: a controlled exposure study of green and conventional office environments," *Environ Health Perspect*, vol. 124, no. 6, pp. 805–812, 2016.

[17] L. Chen, X. Zhou, F. Xiao, Y. Deng, and S. Mahadevan, "Evidential analytic hierarchy process dependence assessment methodology in human reliability analysis," *Nuclear Engineering and Technology*, vol. 49, no. 1, pp. 123–133, 2017.

[18] R. M. Enoka and J. Duchateau, "Translating fatigue to human performance," *Med Sci Sports Exerc*, vol. 48, no. 11, p. 2228, 2016.

[19] X. Deng and W. Jiang, "Dependence assessment in human reliability analysis using an evidential network approach extended by belief rules and uncertainty measures," *Ann Nucl Energy*, vol. 117, pp. 183–193, 2018.

[20] J. Zhao and Y. Deng, "Performer Selection in Human Reliability Analysis: D numbers Approach.," *International Journal of Computers, Communications & Control*, vol. 14, no. 3, 2019.

[21] R. Islam, H. Yu, R. Abbassi, V. Garaniya, and F. Khan, "Development of a monograph for human error likelihood assessment in marine operations," *Saf Sci*, vol. 91, pp. 33–39, 2017.

[22] M. Musharraf, J. Smith, F. Khan, B. Veitch, and S. MacKinnon, "Assessing offshore emergency evacuation behaviour in a virtual environment using a Bayesian Network approach," *Reliab Eng Syst Saf*, vol. 152, pp. 28–37, 2016.

[23] E. Zarei, M. Yazdi, R. Abbassi, and F. Khan, "A hybrid model for human factor analysis in process accidents: FBN-HFACS," *J Loss Prev Process Ind*, vol. 57, pp. 142–155, 2019.

[24] S. Anastasova *et al.*, "A wearable multisensing patch for continuous sweat monitoring," *Biosens Bioelectron*, vol. 93, pp. 139–145, 2017.

[25] T. R. Ray *et al.*, "Bio-integrated wearable systems: a comprehensive review," *Chem Rev*, vol. 119, no. 8, pp. 5461–5533, 2019.

[26] M. Causse, Z. Chua, V. Peysakhovich, N. Del Campo, and N. Matton, "Mental workload and neural efficiency quantified in the prefrontal cortex using fNIRS," *Sci Rep*, vol. 7, no. 1, pp. 1–15, 2017.

[27] P. Aricò *et al.*, "Human-machine interaction assessment by neurophysiological measures: a study on professional air traffic controllers," in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, IEEE, 2018, pp. 4619–4622.

[28] K. Zwirglmaier, D. Straub, and K. M. Groth, "Capturing cognitive causal paths in human reliability analysis with Bayesian network models," *Reliab Eng Syst Saf*, vol. 158, pp. 117–129, 2017.

[29] E. Akyuz, E. Celik, and M. Celik, "A practical application of human reliability assessment for operating procedures of the emergency fire pump at ship," *Ships and Offshore Structures*, vol. 13, no. 2, pp. 208–216, Feb. 2018, doi: 10.1080/17445302.2017.1354658.

[30] K. Mandrick, V. Peysakhovich, F. Rémy, E. Lepron, and M. Causse, "Neural and psychophysiological correlates of human performance under stress and high mental workload," *Biol Psychol*, vol. 121, pp. 62–73, 2016.

[31] W. Kim and J. Park, "Examining structural relationships between work engagement, organizational procedural justice, knowledge sharing, and innovative work behaviour for sustainable organizations," *Sustainability*, vol. 9, no. 2, p. 205, 2017.

[32] V. Muto *et al.*, "Local modulation of human brain responses by circadian rhythmicity and sleep debt," *Science (1979)*, vol. 353, no. 6300, pp. 687–690, 2016.

[33] R. Moura, M. Beer, E. Patelli, J. Lewis, and F. Knoll, "Learning from major accidents to improve system design," *Saf Sci*, vol. 84, pp. 37–45, 2016.

[34] P. Sotiralis, N. P. Ventikos, R. Hamann, P. Golyshev, and A. P. Teixeira, "Incorporation of human factors into ship collision risk models focusing on human centred design aspects," *Reliab Eng Syst Saf*, vol. 156, pp. 210–227, 2016.

[35] F. Zhang, R. de Dear, and P. Hancock, "Effects of moderate thermal environments on cognitive performance: A multidisciplinary review," *Appl Energy*, vol. 236, pp. 760–777, 2019.

[36] F. B. Nerbass, R. Pecoits-Filho, W. F. Clark, J. M. Sontrop, C. W. McIntyre, and L. Moist, "Occupational Heat Stress and Kidney Health: From Farms to Factories," *Kidney Int Rep*, vol. 2, no. 6, pp. 998–1008, 2017, doi: https://doi.org/10.1016/j.ekir.2017.08.012.

[37] R. W. Proctor and K.-P. L. Vu, "Human Information Processing BT - Encyclopedia of the Sciences of Learning," N. M. Seel, Ed., Boston, MA: Springer US, 2012, pp. 1458–1460. doi: 10.1007/978-1-4419-1428-6_722.

[38] B. G. Kanki, "Chapter 2 - Cognitive functions and human error," T. Sgobba, B. Kanki, J.-F. Clervoy, and G. M. B. T.-S. S. and H. P. Sandal, Eds., Butterworth-Heinemann, 2018, pp. 17–52. doi: https://doi.org/10.1016/B978-0-08-101869-9.00002-9.

[39] D. E. Meyer and D. E. Kieras, "A computational theory of executive cognitive processes and multiple-task performance: Part I. Basic mechanisms.," *Psychological Review*, vol. 104, no. 1. American Psychological Association, US, pp. 3–65, 1997. doi: 10.1037/0033-295X.104.1.3.

[40] J. R. Morrow, D. P. Mood, J. G. Disch, and M. Kang, *Measurement and Evaluation in Human Performance*. Human Kinetics, Incorporated, 2015. [Online]. Available: https://books.google.co.in/books?id=WPF6DwAAQBAJ

[41] G. Robert J. Hockey, "Compensatory control in the regulation of human performance under stress and high workload: A cognitive-energetical framework," *Biol Psychol*, vol. 45, no. 1, pp. 73–93, 1997, doi: https://doi.org/10.1016/S0301-0511(96)05223-4.

[42] C. Spence, M. E. R. Nicholls, and J. Driver, "The cost of expecting events in the wrong sensory modality," *Percept Psychophys*, vol. 63, no. 2, pp. 330–336, 2001, doi: 10.3758/BF03194473.

[43] L. W. Way *et al.*, "Causes and prevention of laparoscopic bile duct injuries: analysis of 252 cases from a human factors and cognitive psychology perspective," *Ann Surg*, vol. 237, no. 4, pp. 460–469, Apr. 2003, doi: 10.1097/01.SLA.0000060680.92690.E9.

[44] C. D. Wickens, J. G. Hollands, S. Banbury, and R. Parasuraman, *Engineering psychology and human performance*. Psychology Press, 2015.

[45] R. Parasuraman, T. B. Sheridan, and C. D. Wickens, "A model for types and levels of human interaction with automation," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 30, no. 3, pp. 286–297, 2000, doi: 10.1109/3468.844354.

[46] M. Burnley and A. M. Jones, "Power–duration relationship: Physiology, fatigue, and the limits of human performance," *Eur J Sport Sci*, vol. 18, no. 1, pp. 1–12, 2018.

[47] B. F. Gore, "Chapter 3 - Workload and fatigue," T. Sgobba, B. Kanki, J.-F. Clervoy, and G. M. B. T.-S. S. and H. P. Sandal, Eds., Butterworth-Heinemann, 2018, pp. 53–85. doi: https://doi.org/10.1016/B978-0-08-101869-9.00003-0.

[48] P. Li, L. Zhang, L. Dai, and X.-F. Li, "Study on operator's SA reliability in digital NPPs. Part 1: The analysis method of operator's errors of situation awareness," *Ann Nucl Energy*, vol. 102, pp. 168–178, 2017.

[49] K. Mandrick, V. Peysakhovich, F. Rémy, E. Lepron, and M. Causse, "Neural and psychophysiological correlates of human performance under stress and high mental workload," *Biol Psychol*, vol. 121, pp. 62–73, 2016.

[50] G. Klein, "Developing expertise in decision making," *Think Reason*, vol. 3, no. 4, pp. 337–352, 1997.

[51] J. P. Andersen and H. Gustafsberg, "A training method to improve police use of force decision making: a randomized controlled trial," *Sage Open*, vol. 6, no. 2, p. 2158244016638708, 2016.

[52] B. B. Arnetz, E. Arble, L. Backman, A. Lynch, and A. Lublin, "Assessment of a prevention program for work-related stress among urban police officers," *Int Arch Occup Environ Health*, vol. 86, pp. 79–88, 2013.

[53] by Graham Brent Johnson, "STUDENT PERCEPTIONS OF THE FLIPPED CLASSROOM," 2006.

[54] J. Everly George S, J. M. Lating, J. M. Noel, and J. L. Curtis, "The pharmacological management of stress reactions," *A clinical guide to the treatment of the human stress response*, pp. 317–329, 2013.

[55] K. A. Hine, L. E. Porter, N. J. Westera, G. P. Alpert, and A. Allen, "Exploring police use of force decision-making processes and impairments using a naturalistic decision-making approach," *Crim Justice Behav*, vol. 45, no. 11, pp. 1782–1801, 2018.

[56] K. L. Chrouser, J. Xu, S. Hallbeck, M. B. Weinger, and M. R. Partin, "The influence of stress responses on surgical performance and outcomes: Literature review and the development of the surgical stress effects (SSE) framework," *The American Journal of Surgery*, vol. 216, no. 3, pp. 573–584, 2018, doi: https://doi.org/10.1016/j.amjsurg.2018.02.017.

[57] J. Reason, *Human Error*. Cambridge University Press, 1990. doi: 10.1017/CBO9781139062367.

[58] J. Rasmussen, "Human errors. A taxonomy for describing human malfunction in industrial installations," *Journal of occupational accidents*, vol. 4, no. 2–4, pp. 311–333, 1982.

[59] D. C. McFarlane and K. A. Latorella, "The scope and importance of human interruption in human-computer interaction design," *Hum Comput Interact*, vol. 17, no. 1, pp. 1–61, 2002.

[60] E. Calixto, "Chapter 5 - Human Reliability Analysis," E. B. T.-G. and O. R. E. (Second E. Calixto, Ed., Boston: Gulf Professional Publishing, 2016, pp. 471–552. doi: https://doi.org/10.1016/B978-0-12-805427-7.00005-1.

[61] M. Philippart, "Chapter 12 - Human reliability analysis methods and tools," T. Sgobba, B. Kanki, J.-F. Clervoy, and G. M. B. T.-S. S. and H. P. Sandal, Eds., Butterworth-Heinemann, 2018, pp. 501–568. doi: https://doi.org/10.1016/B978-0-08-101869-9.00012-1.

[62] R. Islam, R. Abbassi, V. Garaniya, and F. Khan, "Development of a human reliability assessment technique for the maintenance procedures of marine and offshore operations," *J Loss Prev Process Ind*, vol. 50, pp. 416–428, Nov. 2017, doi: 10.1016/J.JLP.2017.10.015.

[63] A. D. Swain, "THERP," Sandia Corp., Albuquerque, N. Mex., 1964.

[64] A. R. Kim, J. Park, Y. Kim, J. Kim, and P. H. Seong, "Quantification of performance shaping factors (PSFs)'weightings for human reliability analysis (HRA) of low power and shutdown (LPSD) operations," *Ann Nucl Energy*, vol. 101, pp. 375–382, 2017.

[65] E. Akyuz and M. Celik, "Application of CREAM human reliability model to cargo loading process of LPG tankers," *J Loss Prev Process Ind*, vol. 34, pp. 39–48, 2015.

[66] N. J. Ekanem, A. Mosleh, and S. H. Shen, "Phoenix - A model-based Human Reliability Analysis methodology: Qualitative Analysis Procedure," *Reliab Eng Syst Saf*, vol. 145, pp. 301–315, 2016, doi: 10.1016/j.ress.2015.07.009.

[67] J. Zhao and Y. Deng, "Performer Selection in Human Reliability Analysis: D numbers Approach.," *International Journal of Computers, Communications & Control*, vol. 14, no. 3, 2019.

[68] A. C. Marinescu, S. Sharples, A. C. Ritchie, T. Sanchez Lopez, M. McDowell, and H. P. Morvan, "Physiological parameter response to variation of mental workload," *Hum Factors*, vol. 60, no. 1, pp. 31–56, 2018.

[69] M. R. Nassar, J. C. Helmers, and M. J. Frank, "Chunking as a rational strategy for lossy data compression in visual working memory.," *Psychol Rev*, vol. 125, no. 4, p. 486, 2018.

[70] L. Reinerman-Jones, G. Matthews, and J. E. Mercado, "Detection tasks in nuclear power plant operation: Vigilance decrement and physiological workload monitoring," *Saf Sci*, vol. 88, pp. 97–107, 2016.

[71] E. P. Shaw *et al.*, "Measurement of attentional reserve and mental effort for cognitive workload assessment under various task demands during dual-task walking," *Biol Psychol*, vol. 134, pp. 39–51, 2018.

[72] L. Chen, X. Zhou, F. Xiao, Y. Deng, and S. Mahadevan, "Evidential analytic hierarchy process dependence assessment methodology in human reliability analysis," *Nuclear Engineering and Technology*, vol. 49, no. 1, pp. 123–133, 2017.

[73] M. A. Diaconeasa and A. Mosleh, "Branching rules and quantification based on human behaviour in the ADS-IDAC dynamic PRA platform," in *Safety and Reliability–Safe Societies in a Changing World*, CRC Press, 2018, pp. 1749–1756.

[74] P. Li, L. Zhang, L. Dai, Y. Zou, and X. Li, "An assessment method of operator's situation awareness reliability based on fuzzy logic-AHP," *Saf Sci*, vol. 119, pp. 330–343, 2019.

[75] T. Sakurahara, G. Schumock, S. Reihani, E. Kee, and Z. Mohaghegh, "Simulation-informed probabilistic methodology for common cause failure analysis," *Reliab Eng Syst Saf*, vol. 185, pp. 84–99, 2019.

[76] R. Paprocki and A. Lenskiy, "What Does Eye-Blink Rate Variability Dynamics Tell Us About Cognitive Performance?," *Front Hum Neurosci*, vol. 11, 2017, [Online]. Available: https://www.frontiersin.org/articles/10.3389/fnhum.2017.00620

[77] E. Awh, E. K. Vogel, and S.-H. Oh, "Interactions between attention and working memory," *Neuroscience*, vol. 139, no. 1, pp. 201–208, 2006, doi: https://doi.org/10.1016/j.neuroscience.2005.08.023.

[78] O. E. Krigolson, "Event-related brain potentials and the study of reward processing: Methodological considerations," *International Journal of Psychophysiology*, vol. 132, pp. 175–183, 2018, doi: https://doi.org/10.1016/j.ijpsycho.2017.11.007.

[79] S. Fazli *et al.*, "Enhanced performance by a hybrid NIRS–EEG brain computer interface," *Neuroimage*, vol. 59, no. 1, pp. 519–529, 2012, doi: https://doi.org/10.1016/j.neuroimage.2011.07.084.

[80] S. Aviyente, A. Tootell, and E. M. Bernat, "Time-frequency phase-synchrony approaches with ERPs," *International Journal of Psychophysiology*, vol. 111, pp. 88–97, 2017, doi: https://doi.org/10.1016/j.ijpsycho.2016.11.006.

[81] J. Khan, M. U. G. Khan, R. Iqbal, and O. Riaz, "Robust Multisensor Fusion for the Development of EEG Controlled Vehicle," *IEEE Sens J*, vol. 21, no. 14, pp. 15635–15642, 2021, doi: 10.1109/JSEN.2020.2992714.

[82] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "The German Traffic Sign Recognition Benchmark: A multi-class classification competition," in *The 2011 International Joint Conference on Neural Networks*, 2011, pp. 1453–1460. doi: 10.1109/IJCNN.2011.6033395.

[83] A. Hampshire, S. Sandrone, and P. J. Hellyer, "A Large-Scale, Cross-Sectional Investigation Into the Efficacy of Brain Training," *Front Hum Neurosci*, vol. 13, 2019, [Online]. Available: https://www.frontiersin.org/articles/10.3389/fnhum.2019.00221

[84] J. L. Hardy *et al.*, "Enhancing cognitive abilities with comprehensive training: A large, online, randomized, active-controlled trial," *PLoS One*, vol. 10, no. 9, p. e0134467, 2015.

[85] E. K. S. Louis *et al.*, "The normal EEG," in *Electroencephalography (EEG): An Introductory Text and Atlas of Normal and Abnormal Findings in Adults, Children, and Infants [Internet]*, American Epilepsy Society, 2016.

[86] F. Tadel, S. Baillet, J. C. Mosher, D. Pantazis, and R. M. Leahy, "Brainstorm: A User-Friendly Application for MEG/EEG Analysis," *Comput Intell Neurosci*, vol. 2011, p. 879716, 2011, doi: 10.1155/2011/879716.

[87] G. Niso *et al.*, "MEG-BIDS, the brain imaging data structure extended to magnetoencephalography," *Sci Data*, vol. 5, no. 1, p. 180110, 2018, doi: 10.1038/sdata.2018.110.

[88] R. C. K. Chan, D. Shum, T. Toulopoulou, and E. Y. H. Chen, "Assessment of executive functions: Review of instruments and identification of critical issues," *Archives of Clinical Neuropsychology*, vol. 23, no. 2, pp. 201–216, Mar. 2008, doi: 10.1016/j.acn.2007.08.010.

[89] E. P. Shaw *et al.*, "Measurement of attentional reserve and mental effort for cognitive workload assessment under various task demands during dual-task walking," *Biol Psychol*, vol. 134, pp. 39–51, 2018, doi: https://doi.org/10.1016/j.biopsycho.2018.01.009.

[90] M. U. Iqbal, B. Srinivasan, and R. Srinivasan, "Dynamic assessment of control room operator's cognitive workload using Electroencephalography (EEG)," *Comput Chem Eng*, vol. 141, p. 106726, 2020, doi: https://doi.org/10.1016/j.compchemeng.2020.106726.

[91] L.-W. Ko, O. Komarov, W.-K. Lai, W.-G. Liang, and T.-P. Jung, "Eyeblink recognition improves fatigue prediction from single-channel forehead EEG in a realistic sustained attention task," *J Neural Eng*, vol. 17, no. 3, p. 036015, 2020, doi: 10.1088/1741-2552/ab909f.

[92] T. G. Simpson and K. Rafferty, "EEG Correlates of Driving Performance," *IEEE Trans Hum Mach Syst*, vol. 52, no. 2, pp. 232–247, 2022, doi: 10.1109/THMS.2021.3137032.

[93] J. Seo, T. H. Laine, and K.-A. Sohn, "Machine learning approaches for boredom classification using EEG," *J Ambient Intell Humaniz Comput*, vol. 10, no. 10, pp. 3831–3846, 2019, doi: 10.1007/s12652-019-01196-3.

[94] Y. Zhou, S. Huang, Z. Xu, P. Wang, X. Wu, and D. Zhang, "Cognitive Workload Recognition Using EEG Signals and Machine Learning: A Review," *IEEE Trans*

Cogn Dev Syst, vol. 14, no. 3, pp. 799–818, 2022, doi: 10.1109/TCDS.2021.3090217.

[95] W. Wang, Z. Li, Y. Wang, and F. Chen, "Indexing Cognitive Workload Based on Pupillary Response under Luminance and Emotional Changes," in *Proceedings of the 2013 International Conference on Intelligent User Interfaces*, in IUI '13. New York, NY, USA: Association for Computing Machinery, 2013, pp. 247–256. doi: 10.1145/2449396.2449428.

[96] Caldwell, J. A., Caldwell, J. L., Thompson, L. A., & Lieberman, H. R. (2019). Fatigue and its management in the workplace. Neuroscience & Biobehavioral Reviews, 96, 272–289. https://doi.org/10.1016/j.neubiorev.2018.10.024

[97] Drews, F. A., Rogers, W. P., Talebi, E., & Lee, S. (2020). The experience and management of fatigue: A study of mine haulage operators. Mining, Metallurgy & Exploration, 37(6), 1837–1846. https://doi.org/10.1007/s42461-020-00259-w

[98] Ramakrishnan, P., Balasingam, B., & Biondi, F. (2021). Chapter 2—Cognitive load estimation for adaptive human–machine system automation. In D. Zhang & B. Wei (Eds.), Learning Control (pp. 35–58). Elsevier. https://doi.org/10.1016/B978-0-12-822314-7.00007-9

[99] Hinss, M. F., Brock, A. M., & Roy, R. N. (2022). Cognitive effects of prolonged continuous human-machine interaction: The case for mental state-based adaptive interfaces. Frontiers in Neuroergonomics, 3. https://www.frontiersin.org/articles/10.3389/fnrgo.2022.935092

[100] C. Sennersten, 'Model-based Simulation Training Supporting Military Operational Processes', PhD dissertation, Blekinge Institute of Technology, Karlskrona, 2010.

[101] Land, S. M. (2000). Cognitive requirements for learning with open-ended learning environments. Educational Technology Research and Development, 48(3), 61–78. https://doi.org/10.1007/BF02319858

[102] Cohen, I., Brinkman, W.-P., & Neerincx, M. A. (2015). Modelling environmental and cognitive factors to predict performance in a stressful training scenario on a naval ship simulator. Cognition, Technology & Work, 17(4), 503–519. https://doi.org/10.1007/s10111-015-0325-3

[103] Patel, V., Chesmore, A., Legner, C. M., & Pandey, S. (2022). Trends in workplace wearable technologies and connected□worker solutions for next□generation occupational safety, health, and productivity. Advanced Intelligent Systems, 4(1), 2100099. https://doi.org/10.1002/aisy.202100099

[104] Ahn, C. R., Lee, S., Sun, C., Jebelli, H., Yang, K., & Choi, B. (2019). Wearable sensing technology applications in construction safety and health. Journal of Construction Engineering and Management, 145(11), 03119007. https://doi.org/10.1061/(ASCE)CO.1943-7862.0001708

Appendices

Appendix 1: Sample of the Recorded RAW EEG Data

%OpenBCI Raw EEG Data			
%Number of channels = 8			
%Sample Rate = 250 Hz			
%Board = OpenBCI_GUI$BoardCytonSerial			
Sample Index	**EXG Channel 0**	**EXG Channel 1**	**EXG Channel 2**
0	75257.45186	43059.71927	38257.33522
1	75248.44411	43051.71735	38256.99994
2	75246.23129	43044.52009	38261.67146
3	75256.89307	43065.866	38272.4897
4	75263.21861	43068.4588	38272.75792
5	75258.07771	43057.68526	38263.79487
6	75262.21278	43056.12064	38273.1379
7	75259.95526	43059.20518	38278.72584
8	75272.94162	43076.01369	38289.18645
9	75279.87066	43085.49083	38291.13105
10	75278.39545	43076.86306	38282.50328
11	75271.01937	43062.71441	38277.6306
12	75268.33716	43071.27512	38281.07277
13	75281.52469	43085.35672	38289.58878
14	75279.78126	43081.48987	38280.42457
15	75286.73265	43082.94273	38281.58686
16	75284.00574	43087.65895	38288.96294
17	75267.75602	43070.13518	38279.93283
18	75270.1253	43082.40629	38284.24672
19	75286.44208	43107.552	38298.35067
20	75284.31866	43097.42666	38290.25934
21	75279.13306	43094.65505	38291.44398
22	75285.86093	43106.2556	38304.69856
23	75285.16803	43111.24004	38302.6422
24	75273.03103	43106.30031	38286.616
25	75292.65586	43119.26432	38297.546
26	75301.77537	43120.71718	38309.19126
27	75298.42261	43120.22544	38313.86278
28	75289.97365	43118.72788	38305.99496
29	75306.15631	43140.96786	38319.24955
30	75307.02803	43140.72199	38316.18736
31	75312.90654	43148.63451	38329.509
32	75310.53726	43152.56842	38340.68487

33	75297.57324	43135.33522	38329.41959
34	75305.91044	43134.75408	38334.0017
35	75300.2331	43128.585	38329.75487
36	75298.31085	43126.10395	38332.23591
37	75298.55672	43127.22154	38339.47788
38	75309.37497	43145.86289	38350.89962
39	75319.18738	43159.27394	38354.94528
40	75315.72286	43151.33907	38351.21254
41	75294.71222	43144.92412	38342.31655
42	75289.79484	43147.02519	38344.57408
43	75302.1777	43172.41677	38359.41563
44	75314.47116	43185.0008	38367.68578
45	75301.14952	43169.10871	38354.07357
46	75284.60923	43153.37308	38347.72567
47	75270.66174	43145.54997	38341.0872
48	75285.36919	43164.14662	38353.53712
49	75304.01055	43182.43035	38367.52932
50	75297.77441	43174.98722	38358.16394
51	75277.25551	43151.85316	38345.75872
52	75264.67148	43148.38864	38345.87048
53	75281.36823	43173.82493	38368.71396
54	75305.23989	43194.94733	38389.65754
55	75280.60827	43167.20881	38363.10367
56	75259.37411	43147.29341	38349.24559
57	75261.31871	43148.79097	38357.80631
58	75278.66367	43167.54409	38376.2912
59	75292.65586	43182.63151	38389.52343
60	75267.17487	43163.43137	38366.18821
61	75258.90473	43162.38083	38370.85973
62	75270.92996	43176.6636	38393.79262
63	75279.40128	43189.56056	38410.13174
64	75282.9105	43190.72285	38421.32997
65	75276.24968	43175.68012	38412.63514
66	75262.68217	43165.75595	38410.13174
67	75252.93681	43160.50329	38413.4398
68	75258.50239	43171.83562	38419.65358
69	75268.136	43180.32928	38424.9286
70	75274.88622	43183.97262	38430.7177
71	75265.81142	43176.28362	38437.1997
72	75255.82019	43166.15828	38445.58161
73	75262.41395	43179.52462	38456.48926
74	75278.70837	43195.8861	38470.12382
75	75278.55191	43194.25442	38470.61556
76	75263.62094	43189.00176	38465.20644
77	75249.09231	43179.74814	38462.0772
78	75264.17974	43200.62467	38483.69133
79	75283.13402	43217.45553	38499.49402
80	75275.2215	43207.17373	38493.88373

81	75264.73853	43200.84819	38492.07324
82	75267.26428	43204.29035	38500.38809
83	75268.20305	43210.30297	38500.38809
84	75281.81526	43226.23977	38508.9488
85	75274.97563	43218.97545	38496.7224
86	75263.41978	43203.19512	38487.55818
87	75260.98344	43200.02117	38496.27536
88	75275.04269	43213.72279	38514.53674
89	75287.42555	43226.23977	38525.71261
90	75285.90563	43229.39136	38525.15382
91	75273.09808	43222.55173	38532.28403
92	75259.97761	43209.3642	38536.28499
93	75273.81334	43226.0386	38549.36076
94	75287.47026	43240.65664	38558.8379
95	75280.51886	43229.21255	38547.61732
96	75273.18749	43215.73445	38546.43268
97	75261.96692	43211.64408	38542.9011
98	75272.13696	43230.64306	38559.08377
99	75283.20107	43243.42826	38571.08665
100	75283.93868	43246.26693	38582.24017
101	75263.82211	43227.22324	38582.30723
102	75251.19337	43213.47692	38583.91655
103	75261.1846	43219.84717	38594.31012
104	75280.29535	43224.094	38602.87083
105	75293.37112	43246.69161	38613.97965
106	75283.13402	43232.65472	38605.48599
107	75272.69575	43234.95695	38606.53652
108	75276.24968	43236.16394	38603.04965
109	75286.1068	43246.80337	38605.75421
110	75267.66661	43222.64114	38585.99527
111	75257.27305	43209.69948	38583.04484
112	75260.51405	43212.58285	38587.17991
113	75272.94162	43198.12127	38583.55893
114	75273.58982	43226.26212	38587.55989
115	75269.4771	43232.18533	38576.62989
116	75294.89103	43249.99967	38604.14488
117	75267.62191	43232.40885	38592.45492
118	75261.7434	43230.68776	38588.67748
119	75285.32449	43242.6683	38605.48599
120	75311.43133	43259.61092	38617.01949
121	75278.21663	43234.62167	38591.18087
122	75271.377	43220.69654	38587.35872
123	75312.39245	43279.88395	38628.50829
124	75315.05231	43272.59729	38621.31102
125	75291.82885	43260.08031	38598.35578
126	75291.51592	43252.77129	38599.98746
127	75286.84441	43253.62065	38606.69298
128	75282.8658	43256.68284	38604.07783

129	75286.30797	43248.36799	38593.68427
130	75278.84248	43240.47783	38576.8534
131	75276.54025	43233.52644	38577.34514
132	75299.20492	43249.35147	38606.38006
133	75292.16412	43249.46323	38599.47337
134	75302.84826	43257.15223	38601.41797
135	75307.11744	43258.76156	38599.89805
136	75296.3886	43252.23485	38599.16044
137	75291.53827	43245.99871	38598.84752
138	75302.33417	43262.89663	38609.26343
139	75316.54988	43270.67504	38612.95147
140	75324.30593	43272.03849	38611.72213
141	75314.44881	43261.2426	38606.82709
142	75303.49646	43253.26303	38604.01077
143	75306.44689	43267.38933	38610.58219
144	75324.5518	43274.83246	38615.52192
145	75320.99787	43270.31741	38607.22942
146	75302.46828	43258.20276	38598.64635
147	75294.80163	43256.7499	38601.61914
148	75322.11546	43280.06277	38620.64047
149	75318.87446	43276.6206	38611.20803
150	75310.22433	43267.09876	38599.24985
151	75309.44202	43270.228	38604.77073
152	75295.82981	43254.17945	38593.95249
153	75307.56447	43263.76835	38603.20611
154	75303.36235	43260.5944	38595.04772
155	75304.7258	43263.43307	38594.44423
156	75290.35363	43249.0162	38586.73287
157	75298.60142	43252.10074	38597.90875
158	75322.0484	43278.29698	38617.1089
159	75329.02215	43285.53895	38615.34311
160	75311.20781	43265.6906	38590.8903
161	75304.65875	43261.22025	38587.13521
162	75307.20685	43264.64006	38596.96997
163	75324.14947	43290.45633	38616.72892
164	75335.30299	43299.48643	38617.7571
165	75319.83558	43279.43692	38596.79116
166	75306.581	43270.49622	38593.72897
167	75303.76468	43269.46804	38596.23237
168	75319.81323	43285.13661	38609.03992
169	75323.56832	43281.00154	38603.09435
170	75320.57319	43277.02293	38594.7795
171	75316.46047	43271.92673	38593.99719
172	75308.10092	43265.93647	38590.15269
173	75320.77436	43281.58269	38598.98163
174	75326.83168	43286.47772	38599.80865
175	75321.55667	43283.77316	38592.32081
176	75330.76558	43299.86641	38608.27996

177	75319.90264	43292.6915	38600.97094
178	75337.69463	43313.23276	38616.84067
179	75346.9706	43319.5583	38620.21579
180	75336.42058	43300.55932	38600.81447
181	75324.73061	43295.17255	38594.17601
182	75332.26315	43302.8839	38607.31883
183	75333.55955	43307.26484	38608.63758
184	75330.13974	43306.97427	38601.1274
185	75326.56346	43301.60985	38592.43257
186	75327.77045	43304.1803	38599.76394
187	75315.74521	43300.3358	38598.15462
188	75313.93472	43297.29596	38590.68913
189	75338.07461	43321.05587	38607.74351
190	75340.55565	43325.79444	38607.22942
191	75331.88317	43316.22789	38604.34605
192	75299.18257	43297.7877	38582.77662
193	75304.65875	43306.97427	38588.90099
194	75329.46918	43327.51552	38604.27899
195	75337.9852	43334.01988	38606.62593
196	75308.92793	43305.54376	38583.69304
197	75306.13396	43311.15405	38588.3422
198	75328.441	43338.19966	38606.80474
199	75336.6888	43336.85855	38601.95441
200	75336.24176	43343.36291	38599.13809
201	75318.04744	43323.33575	38586.66582
202	75311.87836	43312.45045	38585.99527
203	75344.13193	43343.34056	38617.57828
204	75347.12706	43366.63107	38621.6463
205	75330.81029	43340.25602	38597.12644
206	75330.43031	43336.81385	38601.37327
207	75321.44491	43336.76914	38602.3791
208	75325.2447	43341.2842	38603.20611
209	75338.92397	43363.25596	38612.37033
210	75325.8035	43342.73706	38600.05451
211	75312.79478	43331.87411	38595.45006
212	75319.29914	43340.34542	38610.40337
213	75331.74906	43355.87989	38623.16622
214	75345.27187	43373.60482	38629.75998
215	75318.29331	43353.68941	38605.06131
216	75319.54501	43352.8624	38612.39268
217	75325.82585	43359.72439	38626.65309
218	75343.39432	43377.71754	38645.13798
219	75357.94531	43405.36665	38661.07478
220	75332.44197	43390.48039	38641.67346
221	75316.88515	43376.73406	38632.62101
222	75301.82008	43367.27928	38623.09916
223	75293.59463	43368.1957	38615.65603
224	75324.28358	43389.00517	38637.73956

225	75326.00466	43396.18008	38639.34888
226	75314.04648	43384.13249	38635.83966
227	75316.19225	43383.86427	38646.34498
228	75325.2	43405.32194	38660.82891
229	75326.25053	43406.7301	38655.57625
230	75312.92889	43395.1072	38638.63363
231	75317.84628	43402.66209	38650.86003
232	75308.25738	43394.68251	38644.80271
233	75315.40994	43399.24227	38648.26723
234	75327.68104	43423.09158	38656.62678
235	75320.99787	43415.46964	38646.0097
236	75304.05525	43403.80203	38635.6832
237	75293.72874	43400.06928	38638.65598
238	75294.17578	43408.71941	38643.72982
239	75308.34679	43418.77769	38647.88725
240	75293.88521	43415.33553	38634.27504
241	75290.6442	43421.52696	38642.67929
242	75278.97659	43404.31612	38641.62876
243	75286.62089	43418.91181	38648.08841
244	75306.46924	43443.69989	38654.01163
245	75293.03584	43432.65813	38631.2799
246	75272.33812	43419.5153	38617.1089
247	75264.2915	43420.67759	38620.93104
248	75287.62672	43450.49482	38646.52379
249	75298.13204	43465.38108	38647.9096
250	75290.42069	43454.09345	38629.91645
251	75278.99894	43449.64545	38624.28381
252	75278.57426	43454.83106	38628.32947
253	75291.47122	43471.21489	38640.37706
254	75284.98921	43464.84464	38628.01655
255	75277.47903	43462.92239	38619.85816
0	75275.08739	43461.02249	38628.01655
1	75275.28856	43461.29071	38633.35861
2	75295.56159	43483.66481	38651.59764
3	75302.96001	43489.20804	38652.11173
4	75297.46149	43486.27996	38644.53449
5	75285.03392	43475.48407	38637.22547
6	75281.6588	43474.56765	38637.13606
7	75281.6588	43483.97773	38639.08066
8	75301.73067	43504.27312	38652.20113
9	75299.65196	43502.44027	38646.43438
10	75282.50817	43486.68229	38639.99708
11	75277.85901	43487.08463	38644.91447
12	75284.34101	43505.72598	38654.50336
13	75288.85606	43512.69972	38655.50919
14	75273.2769	43495.28772	38638.07483
15	75264.55972	43486.34702	38638.61127
16	75256.6025	43484.06714	38639.88532

17	75270.79585	43510.37514	38650.77062
18	75286.24091	43526.06607	38655.97858
19	75280.51886	43521.931	38645.85324
20	75266.99606	43511.58214	38640.71234
21	75262.12338	43513.34793	38647.75314
22	75259.44117	43519.33819	38645.45091
23	75268.58303	43527.27306	38645.74148
24	75260.5364	43523.40621	38634.23033
25	75245.94071	43518.98056	38632.24103
26	75236.48593	43513.6385	38631.30225
27	75270.05825	43546.09323	38659.66662
28	75268.04659	43548.99896	38653.6987
29	75260.35759	43535.52086	38639.43829
30	75258.72591	43531.72106	38642.7687
31	75236.88826	43509.8834	38629.9835
32	75265.72201	43542.71812	38658.63844
33	75276.74142	43563.05821	38666.70742
34	75260.98344	43551.61411	38647.2614
35	75237.49176	43529.86586	38627.83773
36	75240.64335	43538.96302	38636.3761
37	75249.20407	43556.26327	38649.47422
38	75272.89692	43582.14659	38670.77543
39	75260.5364	43564.9134	38650.99414
40	75231.76971	43543.50043	38631.59283
41	75237.29059	43544.90859	38641.96404
42	75261.1399	43561.44888	38656.53737
43	75266.61608	43590.32733	38665.03104
44	75255.37315	43573.18355	38645.24974
45	75251.52865	43560.39835	38641.49465
46	75248.3547	43577.31862	38647.12729
47	75265.54319	43598.50807	38664.20402
48	75273.70158	43602.50903	38675.4246
49	75265.38673	43574.61406	38663.82404
50	75264.60442	43582.12424	38671.46834
51	75273.65688	43599.84918	38693.9989
52	75266.79489	43605.9512	38696.70346
53	75290.21952	43622.17857	38709.06397
54	75278.52956	43613.37198	38698.75982
55	75246.52186	43577.0951	38673.97173
56	75255.82019	43585.3876	38692.30016
57	75257.40716	43592.4731	38697.15049
58	75271.93579	43606.53235	38707.87933
59	75282.01643	43614.02018	38713.06493
60	75268.58303	43605.72769	38716.52945
61	75281.97173	43606.84527	38732.64506
62	75288.36433	43621.73153	38745.43026
63	75291.38181	43624.81607	38747.71014
64	75286.17385	43625.50898	38749.78885

65	75275.2215	43606.91233	38741.96574
66	75265.76671	43601.39145	38745.38556
67	75278.84248	43617.90939	38759.78008
68	75290.33128	43628.99585	38767.02205
69	75293.05819	43622.80442	38766.95499
70	75285.48095	43612.16499	38767.26791
71	75275.64618	43601.57026	38765.36802
72	75298.37791	43630.58283	38792.03365
73	75307.40801	43641.31166	38795.76639
74	75293.37112	43628.57117	38777.39325
75	75270.0806	43605.99591	38759.24364
76	75277.74725	43617.03767	38775.78393
77	75295.7851	43641.49048	38795.92285
78	75312.88419	43655.1921	38806.42817
79	75316.8181	43652.42048	38801.37668
80	75293.92991	43632.37096	38786.17749
81	75280.72003	43617.90939	38783.53998
82	75288.45373	43628.81704	38794.82762
83	75296.88034	43634.42733	38797.33101
84	75312.57126	43648.06189	38809.26684
85	75295.94157	43628.01237	38799.85676
86	75278.4625	43611.91912	38791.67602
87	75290.08541	43634.82966	38807.76928
88	75308.79382	43654.25332	38822.61083
89	75309.8667	43652.35342	38818.90044
90	75293.32641	43632.92976	38804.52827
91	75289.41486	43635.72373	38808.86451
92	75307.9221	43660.35535	38832.35619
93	75324.23888	43667.93259	38842.50389
94	75330.38561	43673.8558	38844.33673
95	75313.46533	43660.17653	38835.23957
96	75321.71313	43668.40198	38850.52816
97	75350.25631	43697.83922	38877.75259
98	75370.84226	43718.87222	38891.9683
99	75347.64115	43692.36305	38859.26769
100	75344.64602	43688.04916	38855.17733
101	75328.15043	43667.50791	38841.3416
102	75349.83162	43679.57785	38854.35031
103	75362.14743	43693.92767	38861.14524
104	75372.94333	43706.71287	38866.21909
105	75365.65666	43698.97916	38860.58645
106	75347.06001	43678.39321	38844.71671
107	75354.99488	43687.55742	38851.37753
108	75355.19604	43689.63613	38841.3863
109	75362.79563	43697.10162	38839.24053
110	75345.62949	43679.44374	38823.14728
111	75352.04445	43686.0375	38835.06076
112	75368.65179	43706.91403	38851.98103

113	75377.19016	43710.64677	38851.15401
114	75373.1892	43713.03841	38843.2862
115	75354.63725	43691.60309	38826.18711
116	75355.08428	43698.1298	38831.99857
117	75367.46715	43713.97718	38839.82168
118	75378.82184	43720.05686	38838.36881
119	75368.47298	43710.91499	38822.36496
120	75362.52741	43713.08311	38822.99081
121	75350.74804	43699.22503	38814.58656
122	75356.93948	43707.40577	38815.36887
123	75369.81408	43726.76238	38823.59431
124	75374.91028	43729.60105	38822.2085
125	75360.35929	43716.01119	38810.31737
126	75351.44095	43705.63998	38808.61864
127	75364.87435	43716.61469	38820.42036
128	75387.76253	43735.85954	38834.81489
129	75384.25331	43728.59522	38821.73912
130	75366.61778	43711.85377	38808.95392
131	75346.07653	43695.29113	38799.38737
132	75369.14353	43724.54956	38825.85184
133	75364.47202	43717.24054	38816.32999
134	75360.56046	43713.93248	38808.95392
135	75355.3078	43713.15017	38813.96071
136	75341.71794	43707.1599	38811.81494
137	75335.41475	43706.77992	38806.91991
138	75362.90739	43728.25995	38825.00247
139	75367.57891	43738.07236	38826.45533
140	75350.18925	43722.15792	38814.81008
141	75335.72767	43698.48742	38802.27075
142	75342.38849	43703.20364	38809.4233
143	75364.18144	43720.99563	38819.77216
144	75356.91713	43721.53207	38807.79163
145	75353.18438	43726.20359	38807.90339
146	75340.22037	43716.7488	38803.81302
147	75338.20872	43722.26968	38807.72457
148	75348.53522	43738.63116	38821.73912
149	75348.17759	43735.16664	38824.48838
150	75326.07172	43716.48058	38808.19396
151	75319.92499	43723.58843	38807.74692
152	75325.46822	43741.67099	38816.10648
153	75343.28256	43759.79826	38826.45533
154	75355.73248	43760.84879	38827.79644
155	75355.48661	43753.33861	38827.26
156	75343.26021	43750.72345	38835.32898
157	75328.79863	43744.12969	38833.02675
158	75344.73543	43760.71468	38836.91595
159	75358.52645	43780.02659	38840.7381
160	75335.4371	43757.18311	38822.38732

161	75331.30203	43753.07038	38822.1638
162	75334.94536	43764.62624	38833.6526
163	75343.14845	43772.56111	38836.98301
164	75362.21449	43799.3832	38854.77499
165	75348.60228	43784.85457	38845.58843
166	75323.65773	43766.63789	38829.58458
167	75333.82777	43785.86039	38842.90622
168	75379.82766	43826.78644	38878.60195
169	75363.82381	43812.19075	38854.99851
170	75350.63629	43805.88756	38849.45528
171	75342.27673	43800.85841	38847.51068
172	75346.18829	43808.99445	38850.86344
173	75327.85986	43782.95467	38817.29112
174	75312.99595	43769.8789	38792.7489
175	75328.21749	43785.83804	38809.11038
176	75336.84526	43807.7651	38827.52822
177	75347.41763	43822.29374	38838.43587
178	75352.33502	43829.22278	38836.20069
179	75352.29031	43831.79323	38831.64094
180	75331.25732	43813.77772	38819.19102
181	75341.29326	43824.9983	38836.35716
182	75347.77526	43838.11877	38843.50972
183	75356.17952	43849.69698	38845.34256
184	75339.79569	43835.03423	38821.78382
185	75325.82585	43823.72425	38813.78189
186	75342.41084	43848.55704	38843.19679
187	75356.82772	43871.84755	38859.20064
188	75357.58768	43870.19353	38850.327
189	75357.43122	43872.80868	38844.87317
190	75368.65179	43881.3694	38859.24534
191	75371.06578	43887.96316	38872.34347
192	75368.51768	43889.19251	38870.93531
193	75361.90156	43879.24598	38859.20064
194	75374.32913	43899.85429	38870.39886
195	75367.46715	43891.53944	38865.72735
196	75339.34865	43865.23144	38845.02963
197	75354.83841	43886.28678	38860.40763
198	75378.53126	43910.69489	38879.38426
199	75377.59249	43913.86883	38875.85269
200	75365.52255	43900.05546	38869.52715
201	75350.14455	43883.44811	38860.09471
202	75356.62655	43893.64051	38864.16273
203	75370.7305	43907.23037	38872.00819
204	75363.33208	43895.76392	38858.03835
205	75350.03279	43884.92333	38850.66227
206	75335.79473	43871.0876	38841.431
207	75352.08915	43891.33828	38859.13358
208	75364.852	43900.16721	38862.41929

209	75361.72275	43897.93204	38849.36587
210	75350.1669	43886.97969	38841.65452
211	75352.93852	43897.21678	38855.19968
212	75363.73441	43916.05931	38867.78371
213	75362.59447	43914.91937	38859.20064
214	75359.44287	43912.30421	38850.19289
215	75354.95017	43908.95145	38850.86344
216	75342.70142	43895.5404	38844.91787
217	75360.94044	43919.90381	38863.67099
218	75379.91707	43943.26138	38877.41731
219	75374.86558	43935.39356	38867.13551
220	75358.43704	43912.1254	38851.26577
221	75349.63046	43903.67644	38848.4718
222	75364.62848	43923.83771	38862.82162
223	75372.13866	43931.88434	38862.03931
224	75382.264	43940.15449	38862.5534
225	75370.66345	43926.09524	38855.73612
226	75364.40496	43925.51409	38856.60784
227	75368.093	43934.83477	38864.36389
228	75377.14545	43944.1778	38868.14134
229	75373.09979	43940.35565	38857.14428
230	75374.68676	43945.16128	38872.70109
231	75369.12118	43946.23416	38901.02075
232	75372.22807	43955.08545	38933.18491
233	75381.30288	43963.02032	38956.81071
234	75378.10658	43964.89787	38965.08085
235	75365.92488	43957.76766	38968.50067
236	75340.64506	43938.07577	38958.39768
237	75351.64211	43952.89498	38976.45789
238	75380.34175	43980.38763	39000.24015

Appendix 2: Cognitive Test Recordings

S. No.		Memory Test			Attention Test		
Readings	**Subjects**	**No. of Fails**	**No. of Success**	**Duration**	**No. of Fails**	**No. of Success**	**Duration**
R1	s1	0	19	97	8	27	120
R2	s2	0	19	97	4	26	120
R3	s1	0	18	97	1	28	120
R4	s2	3	15	97	1	28	120
R5	s1	0	18	97	0	28	120
R6	s2	0	19	97	0	28	120
R7	s1	1	18	97	1	28	120
R8	s2	0	19	97	0	28	120
R9	s1	1	15	97	0	28	120
R10	s2	1	17	97	0	28	120
R11	s1	0	19	97	0	28	120
R12	s2	0	19	97	1	28	120
R13	s1	0	21	116	1	30	120
R14	s2	0	22	116	1	30	120
R15	s2	0	22	116	1	30	120
R16	s1	0	22	116	1	30	120
R17	s1	2	17	116	7	25	120
R18	s2	0	22	116	1	30	120
R19	s2	0	22	116	0	30	120
R20	s1	2	16	116	5	30	120
R21	s1	0	21	116	1	30	120
R22	s2	3	19	126	7	30	120
R23	s1	0	21	116	0	30	120
R24	s2	2	20	126	3	30	120
R25	s1	1	20	116	0	30	120
R26	s2	0	24	126	0	30	120
R27	s3	0	21	116	3	30	120
R28	s2	0	23	126	2	30	120
R29	s1	0	22	116	1	30	120
R30	s3	1	19	116	5	24	120
R31	s2	2	22	122	2	34	120
R32	s3	0	22	116	1	30	120
R33	s2	1	23	122	1	34	120
R34	s3	0	21	116	1	30	120
R35	s1	1	22	126	2	34	120
R36	s2	0	25	122	1	34	120
R37	s3	2	19	116	5	21	120
R38	s2	2	22	122	3	32	120
R39	s3	1	20	116	1	34	120
R40	s4	2	21	122	0	34	120
R41	s3	2	21	122	2	31	120
R42	s4	2	19	122	0	34	120
R43	s3	1	22	122	0	33	120
R44	s5	3	18	122	4	31	120
R45	s4	3	18	122	2	34	120
R46	s3	2	20	122	4	27	120
R47	s4	2	20	122	2	28	120
R48	s3	1	23	122	2	32	120

R49	s5	0	24	122	4	33	120
R50	s5	2	19	122	1	34	120
R51	s4	1	22	122	0	34	120
R52	s3	2	22	122	0	34	120
R53	s4	1	23	122	0	34	120
R54	s3	2	21	122	0	34	120
R55	s4	1	23	122	0	34	120
R56	s3	3	23	122	0	33	120
R57	s4	1	22	122	0	34	120
R58	s3	0	24	122	1	33	120
R59	s5	1	21	122	0	34	120
R60	s4	1	21	122	1	28	120
R61	s3	2	23	122	4	31	120
R62	s3	0	25	122	3	30	120
R63	s4	0	24	122	2	34	120
R64	s3	2	24	140	0	36	120
R65	s4	2	21	140	1	39	120
R66	s5	2	21	140	0	39	120
R67	s4	2	23	140	4	35	120
R68	s3	0	26	140	0	31	120
R69	s5	1	24	140	1	36	120
R70	s4	1	25	140	0	34	120
R71	s3	2	24	140	2	31	120
R72	s3	1	24	140	0	37	120
R73	s4	0	26	140	0	39	120
R74	s3	4	21	140	1	36	120
R75	s4	2	23	140	0	40	120
R76	s5	3	17	140	1	35	120
R77	s4	0	26	140	1	36	120
R78	s3	0	24	140	4	34	120
R79	s5	1	23	140	0	39	120
R80	s4	2	24	140	1	40	120
R81	s3	1	24	140	2	40	120
R82	s3	1	24	140	0	37	120
R83	s4	1	25	140	2	39	120
R84	s5	2	21	140	1	40	120
R85	s4	2	20	140	2	32	120
R86	s3	1	25	140	1	38	120
R87	s5	1	24	140	3	38	120
R88	s4	0	27	140	0	40	120
R89	s3	0	25	140	0	40	120
R90	s5	0	25	140	3	38	120
R91	s4	0	26	140	1	40	120
R92	s3	1	22	140	1	40	120
R93	s3	1	24	140	0	40	120
R94	s4	2	22	140	1	40	120
R95	s4	1	25	140	1	39	120
R96	s6	0	27	140	2	38	120
R97	s6	1	25	140	0	40	120
R98	s4	0	26	140	3	40	120
R99	s5	1	24	140	3	37	120
R100	s6	3	21	140	4	34	120
R101	s4	2	22	140	3	39	120

R102	s5	0	26	140	3	38	120
R103	s6	1	26	140	0	40	120
R104	s4	1	26	140	5	33	120
R105	s4	2	22	140	0	40	120
R106	s6	1	24	140	1	39	120
R107	s6	2	21	140	4	40	120
R108	s4	1	24	140	3	37	120
R109	s5	1	24	140	2	39	120
R110	s5	1	25	140	4	35	120
R111	s6	0	25	140	0	40	120
R112	s4	1	24	140	0	40	120
R113	s6	1	24	140	3	37	120
R114	s4	0	27	140	5	28	120
R115	s6	2	27	140	1	41	120
R116	s7	1	27	140	0	42	120
R117	s5	5	19	140	0	43	120
R118	s6	3	21	140	2	36	120
R119	s7	3	20	140	0	44	120
R120	s6	5	19	140	0	42	120
R121	s6	3	22	140	1	40	120
R122	s7	1	24	140	1	39	120
R123	s5	0	26	140	2	44	120
R124	s6	3	22	140	1	43	120
R125	s7	1	27	140	3	39	120
R126	s6	3	20	140	0	41	120
R127	s7	0	28	140	2	39	120
R128	s5	2	28	140	3	37	120
R129	s6	2	24	140	2	41	120
R130	s7	1	26	140	1	40	120
R131	s6	3	24	140	0	44	120
R132	s7	1	26	140	1	44	120
R133	s6	3	22	140	8	33	120
R134	s6	4	19	140	3	38	120
R135	s7	2	23	140	2	44	120
R136	s6	2	24	140	2	40	120
R137	s7	2	24	140	4	41	120
R138	s5	3	23	140	1	40	120
R139	s5	2	26	140	1	42	120
R140	s6	2	25	140	3	44	120
R141	s7	3	22	140	2	41	120
R142	s5	1	26	140	2	40	120
R143	s5	2	25	140	2	42	120
R144	s6	3	22	140	4	41	120
R145	s7	2	24	140	2	41	120
R146	s5	4	19	140	4	37	120
R147	s5	3	24	140	4	35	120
R148	s6	1	25	140	1	44	120
R149	s7	3	21	140	3	38	120
R150	s7	2	25	140	0	44	120
R151	s6	1	27	140	1	43	120
R152	s7	3	21	140	2	41	120
R153	s6	0	27	140	3	44	120
R154	s5	2	26	140	6	42	120

R155	s5	3	21	140	4	39	120
R156	s7	2	23	140	3	40	120
R157	s6	2	25	140	0	44	120
R158	s5	0	27	140	0	43	120
R159	s7	2	24	140	2	44	120
R160	s6	4	18	140	2	43	120
R161	s5	3	24	140	3	40	120
R162	s7	1	26	140	0	44	120
R163	s6	3	22	140	4	43	120
R164	s6	3	23	140	0	44	120
R165	s7	2	24	140	1	44	120
R166	s7	1	26	140	3	42	120
R167	s6	2	24	140	0	44	120
R168	s7	0	28	140	2	44	120
R169	s8	1	25	140	4	39	120
R170	s5	1	24	140	4	42	120
R171	s7	2	25	140	2	43	120
R172	s8	1	26	140	1	48	120
R173	s5	0	28	140	1	48	120
R174	s7	0	27	140	2	48	120
R175	s8	1	26	140	2	47	120
R176	s8	3	24	140	0	48	120
R177	s7	0	28	140	3	47	120
R178	s7	0	28	140	3	48	120
R179	s8	0	27	140	1	48	120
R180	s9	3	22	140	3	48	120
R181	s8	1	25	140	1	43	120
R182	s7	1	25	140	1	48	120
R183	s9	0	28	140	1	48	120
R184	s7	2	25	140	5	39	120
R185	s8	1	27	140	2	47	120
R186	s8	0	28	140	0	48	120
R187	s7	0	29	140	2	48	120
R188	s9	4	23	140	3	46	120
R189	s9	0	28	140	0	48	120
R190	s7	2	24	140	1	48	120
R191	s8	0	28	140	0	48	120
R192	s8	1	26	140	5	45	120
R193	s7	0	28	140	0	48	120
R194	s9	0	28	140	5	45	120
R195	s10	1	24	140	0	48	120
R196	s8	1	25	140	6	47	120
R197	s9	2	25	140	4	44	120
R198	s9	1	27	140	3	48	120
R199	s10	0	28	140	2	45	120
R200	s8	0	28	140	2	48	120
R201	s10	3	22	140	0	48	120
R202	s8	2	24	140	3	47	120
R203	s8	2	23	140	1	48	120
R204	s10	1	26	140	2	48	120
R205	s9	0	28	140	1	48	120
R206	s10	1	27	140	1	48	120
R207	s8	0	27	140	7	40	120

R208	s9	3	21	140	2	48	120
R209	s8	1	25	140	1	48	120
R210	s10	3	22	140	2	48	120
R211	s10	2	22	140	1	45	120
R212	s8	2	25	140	0	48	120
R213	s8	1	26	140	2	45	120
R214	s10	2	22	140	2	48	120
R215	s9	3	24	140	1	48	120
R216	s8	1	26	140	3	42	120
R217	s10	0	29	140	1	48	120
R218	s9	3	23	140	2	48	120
R219	s9	0	28	140	3	47	120
R220	s10	1	25	140	1	48	120
R221	s8	1	26	140	1	48	120
R222	s9	1	26	140	0	48	120
R223	s10	1	25	140	1	48	120
R224	s8	4	20	140	3	48	120
R225	s9	0	27	140	0	48	120
R226	s9	0	28	140	2	47	120
R227	s10	1	26	140	2	48	120
R228	s8	1	27	140	0	48	120
R229	s9	0	29	140	0	50	120
R230	s10	0	28	140	1	48	120
R231	s9	1	26	140	3	51	120
R232	s9	1	27	140	3	51	120
R233	s9	2	24	140	3	51	120
R234	s10	1	27	140	6	46	120
R235	s8	2	26	140	3	46	120
R236	s9	3	23	140	8	50	120

Appendix 3: Brainstorm MATLAB Code:

```
% Input files
sFiles = {...
    'NewSubject/@rawR1/data_0raw_R1.mat'};

% Start a new report
bst_report('Start', sFiles);

% Process: Set bad channels
sFiles = bst_process('CallProcess', 'process_channel_setbad', sFiles,
[], ...
    'sensortypes', 'E04, E05, E06, E07, E08, E09, E10, E11, E12, E13,
E14, E15, E16, E17, E18, E19, E20, E21, E22, E23, E24, E25, E26, E27,
E28, E29, E30');

% Process: Band-pass:0.1Hz-49Hz
sFiles = bst_process('CallProcess', 'process_bandpass', sFiles, [],
...
    'sensortypes', 'EEG', ...
    'highpass',    0.1, ...
    'lowpass',     49, ...
    'tranband',    0, ...
    'attenuation', 'strict', ...  % 60dB
    'ver',         '2019', ...  % 2019
    'mirror',      0, ...
    'read_all',    0);

% Process: Detect eye blinks
sFiles = bst_process('CallProcess', 'process_evt_detect_eog', sFiles,
[], ...
    'channelname', 'E01', ...
    'timewindow',  [], ...
    'eventname',   'B');

% Process: SSP EOG: B
sFiles = bst_process('CallProcess', 'process_ssp_eog', sFiles, [], ...
    'eventname',   'B', ...
    'sensortypes', 'EEG', ...
    'usessp',      1, ...
    'select',      1);

% Process: Power spectrum density (Welch)
sFiles = bst_process('CallProcess', 'process_psd', sFiles, [], ...
    'timewindow',  [], ...
    'win_length',  1, ...
    'win_overlap', 50, ...
    'units',       'physical', ...  % Physical: U2/Hz
    'sensortypes', 'EEG', ...
    'win_std',     1, ...
    'edit',        struct(...
         'Comment',         'Power,FreqBands', ...
         'TimeBands',       [], ...
         'Freqs',           {{'delta', '0.1, 4', 'mean'; 'theta', '4,
8', 'mean'; 'alpha', '8, 13', 'mean'; 'beta', '13, 30', 'mean';
'gamma', '30, 49', 'mean'}}, ...
         'ClusterFuncTime', 'none', ...
         'Measure',         'power', ...
         'Output',          'all', ...
         'SaveKernel',      0));
```

```
% Save and display report
ReportFile = bst_report('Save', sFiles);
bst_report('Open', ReportFile);
% bst_report('Export', ReportFile, ExportDir);
```

Appendix 4: Brainstorm Output Sample

		Average power contained in the frequency interval				
		0.1-4	**4-8.**	**8-13.**	**13-30**	**30-49**
		Delta	**Theta**	**Alpha**	**Beta**	**Gamma**
Channel 1	Fp1	5.16E-10	6.73E-11	6.14E-11	4.59E-11	2.50E-11
Channel 2	Fp2	1.49E-08	2.36E-09	1.97E-09	1.38E-09	7.47E-10
Channel 3	FpZ	8.70E-09	6.99E-10	6.62E-10	5.10E-10	2.94E-10
Average		8.04E-09	1.04E-09	8.98E-10	6.45E-10	3.55E-10

Appendix 5: BLINK Python Code:

```
from PyQt5.QtCore import *
from PyQt5.QtGui import *
from PyQt5.QtWidgets import *
import sys
from configparser import ConfigParser
import pandas as pd

import os
import numpy as np
from scipy.signal import *
import matplotlib
import matplotlib.pyplot as plt

from scipy.cluster.hierarchy import linkage
from scipy.cluster.hierarchy import fcluster


from scipy.interpolate import interp1d


class App(QWidget):

    def __init__(self):
        super().__init__()

        self.title = 'Blinks'
        self.width = 324
        self.height = 330
        self.initUI()
        self.filename = str()

        # Destination Folder Path
        self.dump_folder = "./Dump/"

        # Algorithm Parameteres

        self.running_std = None
        self.data_len = None

        # data_path = os.getcwd()
        self.mode = True  # mode True means EEG-IO, otherwise v/r
(EEG-VV or EEG-VR) data
        self.data_path = './data/EEG-VV/'  # or replace w/ EEG-VR
        self.file_idx = 1

        self.fs = 250.0
        self.chan_id = 1

        self.flag_soft = True  # if True, consider soft blinks as
ground truth

        self.blink_len_max = 2.0  # in seconds
        self.blink_len_min = 0.2  # in seconds

        self.delta_init = 100  # in uvolts

        self.corr_threshold_1 = 0.2
        self.corr_threshold_2 = 0.7
```

```
        self.std_threshold_window = int(5 * self.fs)  # %  in seconds
- for each direction

    def initUI(self):
        self.setWindowTitle(self.title)
        self.resize(324, 330)
        self.setFixedSize(self.width, self.height)

        self.select_file = QPushButton("Select File", self)
        self.select_file.clicked.connect(self.openFileNameDialog)
        self.select_file.setGeometry(QRect(90, 40, 141, 41))
        # self.select_file.resize(170, 40)
        # self.select_file.move(95, 35)

        self.file_name = QLabel(self)
        self.file_name.setAlignment(Qt.AlignCenter)
        self.file_name.setGeometry(QRect(90, 100, 141, 20))
        # self.file_name.resize(170, 40)
        # self.file_name.move(95, 100)

        self.progressBar = QProgressBar(self)
        self.progressBar.setGeometry(QRect(90, 160, 141, 31))
        self.progressBar.setProperty("value", 0)
        self.progressBar.setObjectName("progressBar")

        self.upload_btn = QPushButton("Upload", self)
        self.upload_btn.clicked.connect(self.generate_blinks)
        self.upload_btn.setGeometry(QRect(90, 220, 141, 41))
        # self.upload_btn.resize(170, 40)
        # self.upload_btn.move(95, 250)

        self.show()

    @staticmethod
    def lowpass(sig, fc, fs, butter_filt_order):
        B, A = butter(butter_filt_order, np.array(fc) / (fs / 2),
btype='low')
        return lfilter(B, A, sig, axis=0)

    def compute_running_std(self, data_sig, chan_id, fs):
        # Find running std
        std_length = int(0.5 * fs)  # in seconds

        running_std = np.zeros([self.data_len, 1])
        idx = 0
        while (idx < len(data_sig) - std_length):
            running_std[idx] = np.std(data_sig[idx:(idx + std_length),
chan_id])
            idx = idx + 1
        running_std[idx:-1] = running_std[idx - 1]

        # fixing the corrupted signal's std
        for idx in range(self.data_len):
            if running_std[idx] < 1:
                l_index_lhs = max(0, idx - std_length)
                l_index_rhs = max(0, (idx - std_length - 2 *
self.std_threshold_window - int(fs)))
                r_index_lhs = min(self.data_len, idx + std_length)
                r_index_rhs = max(0, idx - std_length - int(fs))
                running_std[l_index_lhs:r_index_lhs] =
min(running_std[l_index_rhs:r_index_rhs])
```

```
                idx = idx + std_length - 1

        return running_std

    @staticmethod
    def args_init(delta_uV):
        args = {}
        args['mintab'], args['maxtab'] = [], []
        args['mn'], args['mx'] = float("inf"), -1 * float("inf")
        args['mnpos'], args['mxpos'] = None, None
        args['min_left'], args['min_right'] = [], []
        args['lookformax'] = True
        args['delta'] = delta_uV
        return args

    @staticmethod
    def peakdet(time, value, args):
        foundMin = False
        if value > args['mx']:
            args['mx'] = value
            args['mxpos'] = time
        if value < args['mn']:
            args['mn'] = value
            args['mnpos'] = time
        if args['lookformax']:
            if value < args['mx'] - args['delta']:
                args['maxtab'].append([args['mxpos'], args['mx']])
                args['mn'] = value
                args['mnpos'] = time
                args['lookformax'] = False
        else:
            if value > args['mn'] + args['delta']:
                args['mintab'].append([args['mnpos'], args['mn']])
                args['min_left'].append([-1, -1])
                args['min_right'].append([-1, -1])
                args['mx'] = value
                args['mxpos'] = time
                args['lookformax'] = True
                foundMin = True
        return foundMin

    def find_expoints(self, stat_min2, data_sig, chan_id):
        # Parameters
        offset_t = 0.00  # in seconds
        win_size = 25
        win_offset = 10
        search_maxlen_t = 1.5  # in seconds

        offset_f = int(offset_t * self.fs)
        search_maxlen_f = int(search_maxlen_t * self.fs)
        iters = int(search_maxlen_f / win_offset)

        data_len = len(data_sig)
        p_blinks_t, p_blinks_val = [], []
        for idx in range(len(stat_min2)):
            # x_indR and x_indL are starting points for left and right
window
            x_indR = int(self.fs * stat_min2[idx, 0]) + offset_f
            x_indL = int(self.fs * stat_min2[idx, 0]) - offset_f
            start_index = max(0, int(self.fs * stat_min2[idx, 0]) -
self.std_threshold_window)
```

```
            end_index = min(int(self.fs * stat_min2[idx, 0]) +
self.std_threshold_window, data_len)
            stable_threshold = 2 *
min(self.running_std[start_index:end_index])
            min_val = stat_min2[idx, 1]
            max_val = min_val
            found1, found2 = 0, 0
            state1, state2 = 0, 0

            for iter in range(iters):
                if (x_indR + win_size > data_len):
                    x_indR = x_indR - (x_indR + win_size - data_len)
                if (x_indL < 0):
                    x_indL = 0
                if (np.std(data_sig[x_indR:x_indR + win_size,
chan_id]) < stable_threshold) and state1 == 1 and \
                        data_sig[x_indR, chan_id] > min_val:
                    found1 = 1
                    max_val = max(data_sig[x_indR, chan_id], max_val)
                if (np.std(data_sig[x_indL:x_indL + win_size,
chan_id]) < stable_threshold) and state2 == 1 and \
                        data_sig[x_indL + win_size, chan_id] >
min_val:
                    found2 = 1
                    max_val = max(data_sig[x_indL + win_size,
chan_id], max_val)
                if (np.std(data_sig[x_indR:x_indR + win_size,
chan_id]) > 2.5 * stable_threshold) and state1 == 0:
                    state1 = 1
                if (np.std(data_sig[x_indL:x_indL + win_size,
chan_id]) > 2.5 * stable_threshold) and state2 == 0:
                    state2 = 1
                if (found1 == 1) and data_sig[x_indR, chan_id] <
(max_val + 2 * min_val) / 3:
                    found1 = 0
                if (found2 == 1) and data_sig[x_indL + win_size,
chan_id] < (max_val + 2 * min_val) / 3:
                    found2 = 0
                if (found1 == 0):
                    x_indR = x_indR + win_offset
                if (found2 == 0):
                    x_indL = x_indL - win_offset
                if found1 == 1 and found2 == 1:
                    break
            if found1 == 1 and found2 == 1:
                if (x_indL + win_size) / self.fs > stat_min2[idx, 0]:
                    p_blinks_t.append([(x_indL) / self.fs,
stat_min2[idx, 0], x_indR / self.fs])
                    p_blinks_val.append([data_sig[x_indL, chan_id],
stat_min2[idx, 1], data_sig[x_indR, chan_id]])
                else:
                    p_blinks_t.append([(x_indL + win_size) / self.fs,
stat_min2[idx, 0], x_indR / self.fs])
                    p_blinks_val.append(
                        [data_sig[x_indL + win_size, chan_id],
stat_min2[idx, 1], data_sig[x_indR, chan_id]])

        p_blinks_t = np.array(p_blinks_t)
        p_blinks_val = np.array(p_blinks_val)

        return p_blinks_t, p_blinks_val
```

```
    @staticmethod
    def compute_correlation(p_blinks_t, data_sig, chan_id, fs):
        total_p_blinks = len(p_blinks_t)
        corr_matrix = np.ones([total_p_blinks, total_p_blinks])
        pow_matrix = np.ones([total_p_blinks, total_p_blinks])
        for idx_i in range(total_p_blinks):
            for idx_j in range(idx_i + 1, total_p_blinks):

                blink_i_left = data_sig[int(fs * p_blinks_t[idx_i,
0]):int(fs * p_blinks_t[idx_i, 1]), chan_id]
                blink_i_right = data_sig[int(fs * p_blinks_t[idx_i,
1]):int(fs * p_blinks_t[idx_i, 2]), chan_id]

                blink_j_left = data_sig[int(fs * p_blinks_t[idx_j,
0]):int(fs * p_blinks_t[idx_j, 1]), chan_id]
                blink_j_right = data_sig[int(fs * p_blinks_t[idx_j,
1]):int(fs * p_blinks_t[idx_j, 2]), chan_id]

                left_interp = interp1d(np.arange(blink_i_left.size),
blink_i_left)
                compress_left = left_interp(np.linspace(0,
blink_i_left.size - 1, blink_j_left.size))
                right_interp = interp1d(np.arange(blink_i_right.size),
blink_i_right)
                compress_right = right_interp(np.linspace(0,
blink_i_right.size - 1, blink_j_right.size))

                sigA = np.concatenate((compress_left, compress_right))
                sigB = np.concatenate((blink_j_left, blink_j_right))

                corr = np.corrcoef(sigA, sigB)[0, 1]
                corr_matrix[idx_i, idx_j] = corr
                corr_matrix[idx_j, idx_i] = corr

                if np.std(sigA) > np.std(sigB):
                    pow_ratio = np.std(sigA) / np.std(sigB)
                else:
                    pow_ratio = np.std(sigB) / np.std(sigA)

                pow_matrix[idx_i, idx_j] = pow_ratio
                pow_matrix[idx_j, idx_i] = pow_ratio

        return corr_matrix, pow_matrix

    def generate_blinks(self):
        source = self.filename
        file_sig = self.file_name.text()
        print("File Name: ", file_sig)

        # Loading Data

        data_sig = np.loadtxt(open(source, "rb"), delimiter=",",
skiprows=5,
                              usecols=(0, 1, 2))
        data_sig = data_sig[0:(int(300 * self.fs) + 1), :]
        data_sig = data_sig[:, 0:3]
        data_sig[:, 0] = np.array(range(0, len(data_sig))) / self.fs

        # Step1: Low Pass Filter
        data_sig[:, 1] = self.lowpass(data_sig[:, 1], 10, self.fs, 4)
```

```
        data_sig[:, 2] = self.lowpass(data_sig[:, 2], 10, self.fs, 4)

        time_min = data_sig[0, 0]
        time_max = data_sig[-1, 0]

        self.data_len = len(data_sig)

        ##################################################

        args_chan1 = self.args_init(self.delta_init)

        self.running_std = self.compute_running_std(data_sig,
self.chan_id, self.fs)
        for idx in range(len(data_sig[:, 0])):
            self.peakdet(data_sig[idx, 0], data_sig[idx,
self.chan_id], args_chan1)

        min_pts = np.array(args_chan1['mintab'])
        p_blinks_t, p_blinks_val = self.find_expoints(min_pts,
data_sig, self.chan_id)
        corr_matrix, pow_matrix = self.compute_correlation(p_blinks_t,
data_sig, self.chan_id, self.fs)

        # fingerprint
        blink_fp_idx = np.argmax(sum(corr_matrix))
        t = corr_matrix[blink_fp_idx, :] > self.corr_threshold_1
        blink_index = [i for i, x in enumerate(t) if x]

        blink_template_corrmat = corr_matrix[np.ix_(blink_index,
blink_index)]
        blink_template_powmat = pow_matrix[np.ix_(blink_index,
blink_index)]
        blink_templates_corrWpower = blink_template_corrmat /
blink_template_powmat

        blink_var = []
        for idx in blink_index:
            blink_var.append(np.var(data_sig[int(self.fs *
p_blinks_t[idx, 0]):int(self.fs * p_blinks_t[idx, 2]), self.chan_id]))

        Z = linkage(blink_templates_corrWpower, 'complete',
'correlation')
        groups = fcluster(Z, 2, 'maxclust')

        grp_1_blinks_var = [blink_var[i] for i, x in enumerate(groups
== 1) if x]
        grp_2_blinks_var = [blink_var[i] for i, x in enumerate(groups
== 2) if x]
        if np.mean(grp_1_blinks_var) > np.mean(grp_2_blinks_var):
            selected_group = 1
        else:
            selected_group = 2
        template_blink_idx = [blink_index[i] for i, x in
enumerate(groups == selected_group) if x]

        # computing delta new
        delta_new = 0
        for idx in template_blink_idx:
            delta_new = delta_new + min(p_blinks_val[idx, 0],
p_blinks_val[idx, 2]) - p_blinks_val[idx, 1]
        delta_new = delta_new / len(template_blink_idx)
```

```
        # 2nd pass

        args_chan1 = self.args_init(delta_new / 3.0)

        for idx in range(len(data_sig[:, 0])):
            self.peakdet(data_sig[idx, 0], data_sig[idx,
self.chan_id], args_chan1)

        min_pts = np.array(args_chan1['mintab'])
        p_blinks_t, p_blinks_val = self.find_expoints(min_pts,
data_sig, self.chan_id)
        corr_matrix, pow_matrix = self.compute_correlation(p_blinks_t,
data_sig, self.chan_id, self.fs)

        s_fc = (sum(corr_matrix))
        sort_idx = sorted(range(len(s_fc)), key=lambda k: s_fc[k])

        t = corr_matrix[sort_idx[-1], :] > self.corr_threshold_2
        blink_index1 = set([i for i, x in enumerate(t) if x])
        t = corr_matrix[sort_idx[-2], :] > self.corr_threshold_2
        blink_index2 = set([i for i, x in enumerate(t) if x])
        t = corr_matrix[sort_idx[-3], :] > self.corr_threshold_2
        blink_index3 = set([i for i, x in enumerate(t) if x])

        blink_index =
list(blink_index1.union(blink_index2).union(blink_index3))

        blink_template_corrmat = corr_matrix[np.ix_(blink_index,
blink_index)]
        blink_template_powmat = pow_matrix[np.ix_(blink_index,
blink_index)]
        blink_templates_corrWpower = blink_template_corrmat /
blink_template_powmat

        blink_var = []
        for idx in blink_index:
            blink_var.append(np.var(data_sig[int(self.fs *
p_blinks_t[idx, 0]):int(self.fs * p_blinks_t[idx, 2]), self.chan_id]))

        Z = linkage(blink_templates_corrWpower, 'complete',
'correlation')
        groups = fcluster(Z, 2, 'maxclust')

        grp_1_blinks_var = [blink_var[i] for i, x in enumerate(groups
== 1) if x]
        grp_2_blinks_var = [blink_var[i] for i, x in enumerate(groups
== 2) if x]

        if np.mean(grp_1_blinks_var) > np.mean(grp_2_blinks_var) and
np.mean(grp_1_blinks_var) / np.mean(
                grp_2_blinks_var) > 10:
            blink_index = [blink_index[i] for i, x in enumerate(groups
== 1) if x]
        elif np.mean(grp_2_blinks_var) > np.mean(grp_1_blinks_var) and
np.mean(grp_2_blinks_var) / np.mean(
                grp_1_blinks_var) > 10:
            blink_index = [blink_index[i] for i, x in enumerate(groups
== 2) if x]

        final_blinks_t = p_blinks_t[blink_index, :]
```

```
        final_blinks_val = p_blinks_val[blink_index, :]

        print("Final Blinks:")
        # print (final_blinks_t)

        final_df = pd.DataFrame(final_blinks_t, columns=["Final
Blinks", "", ""])
        print(final_df)

        destination =
f"{self.dump_folder}output_{self.file_name.text()}"
        print("****** Blinks Generated ******")

        final_df.to_csv(destination, index=False)
        self.progressBar.setValue(100)

    def openFileNameDialog(self):
        options = QFileDialog.Options()
        options |= QFileDialog.DontUseNativeDialog
        self.filename, _ = QFileDialog.getOpenFileName(self, "Upload a
file", "",
                                                       "All Files
(*);;Python Files (*.py)", options=options)
        if self.filename:
            name = self.filename.split("/")[-1]
            self.file_name.setText(name)
            self.progressBar.setValue(0)


if __name__ == '__main__':
    app = QApplication(sys.argv)
    app.setStyle("Fusion")
    ex = App()
    sys.exit(app.exec_())
```

Appendix 6: BLINK Output Sample

Blink Time Stamps		
Start	**Minima**	**End**
2.932	2.992	3.192
8.228	8.408	8.648
13.4	13.5	13.66
44.692	44.832	44.912
48.012	48.072	48.152
75.036	75.096	75.296
81.216	81.276	81.396
83.34	83.44	83.56
88.052	88.112	88.192
103.404	103.464	103.584
141.28	141.34	141.46
157.408	157.548	157.828
159.564	159.664	159.944
160.992	161.092	161.332
167.168	167.228	167.388
168.168	168.228	168.388
171.108	171.168	171.288
184.984	185.044	185.164
191.772	191.872	192.032
200.396	200.456	200.616
209.516	209.576	209.776
212.072	212.172	212.372
218.924	219.024	219.224
222.712	222.812	223.012
225.672	225.732	225.972
244.044	244.144	244.304
246.584	246.684	246.844
248.384	248.444	248.644
249.388	249.488	249.728
251.084	251.144	251.264
270.032	270.092	270.212
274.164	274.224	274.384

Appendix 7: Data

S. No.	Subject ID	Reaction time for each attempt in attention game	Avg. blink duration in attention game	Avg. blink interval in attention game	Reaction time for each attempt in memory game	Avg. blink duration in memory game	Avg. blink interval in memory game	Avg. delta	Avg. theta	Avg. alpha	Avg. beta	Avg. gamma	Attention Success Rate	Memory Success Rate
1	1	3.428571	0.25125	8.749419	5.105263	0.25125	8.749419	8.04E-09	1.04E-09	8.98E-10	6.45E-10	3.55E-10	0.771429	1
2	2	4	0.264444	12.69	5.105263	0.264444	12.69	2.36E-08	5.06E-09	4.83E-09	3.65E-09	1.77E-09	0.866667	1
3	1	4.137931	0.276	12.89467	5.388889	0.276	12.89467	1.55E-08	2.81E-09	2.74E-09	2.07E-09	1.03E-09	0.965517	1
4	2	4.137931			5.388889			1.92E-08	3.99E-09	3.83E-09	2.9E-09	1.39E-09	0.965517	0.833333
5	1	4.285714			5.388889			4.16E-08	3.71E-08	3.54E-08	2.96E-08	1.77E-08	1	1
6	2	4.285714	0.281818	23.0268	5.105263	0.281818	23.0268	1.92E-08	3.88E-09	3.69E-09	2.79E-09	1.33E-09	1	1
7	1	4.137931			5.105263			9.17E-09	9.25E-09	9.23E-09	9.22E-09	8.98E-09	0.965517	0.947368
8	2	4.285714			5.105263			2.43E-08	5.3E-09	5.05E-09	3.81E-09	1.76E-09	1	1
9	1	4.285714	0.238182	20.8408	6.0625	0.238182	20.8408	9.91E-09	1.98E-09	1.9E-09	1.44E-09	7.11E-10	1	0.9375
10	2	4.285714	0.264444	10.9665	5.388889	0.264444	10.9665	2.26E-08	4.9E-09	4.68E-09	3.54E-09	1.66E-09	1	0.944444
11	1	4.285714	0.41	28.75067	5.105263	0.41	28.75067	2.31E-08	4.99E-09	4.76E-09	3.6E-09	1.69E-09	1	1
12	2	4.137931			5.105263			2.18E-08	4.55E-09	4.37E-09	3.3E-09	1.6E-09	0.965517	1
13	1	3.870968	0.58	12.95429	5.52381	0.58	12.95429	3.18E-08	6.92E-09	6.61E-09	4.98E-09	2.3E-09	0.967742	1
14	2	3.870968	0.228235	8.5745	5.272727	0.228235	8.5745	2.49E-08	5.19E-09	4.91E-09	3.78E-09	1.84E-09	0.967742	1
15	2	3.870968	0.336923	11.57184	5.272727	0.336923	11.57184	1.09E-08	2.27E-09	2.18E-09	1.66E-09	7.97E-10	0.967742	1
16	1	3.870968	0.342963	9.800308	5.272727	0.342963	9.800308	1.73E-08	3.39E-09	3.24E-09	2.45E-09	1.14E-09	0.967742	1
17	1	3.75	0.34	10.2513	6.105263	0.34	10.2513	5.97E-09	1.13E-09	1.08E-09	8.22E-10	4.04E-10	0.78125	0.894737
18	2	3.870968	0.309524	7.036683	5.272727	0.309524	7.036683	1.38E-08	2.93E-09	2.81E-09	2.12E-09	1.01E-09	0.967742	1
19	2	4	0.383556	6.440455	5.272727	0.383556	6.440455	6E-09	7.48E-10	7.06E-10	5.41E-10	2.83E-10	1	1
20	1	3.428571	0.3128	5.714286	6.444444	0.3128	5.714286	7.68E-09	1.29E-09	1.21E-09	9.1E-10	4.51E-10	0.857143	0.888889
21	1	3.870968	0.314857	8.539176	5.52381	0.314857	8.539176	9.01E-09	1.67E-09	1.61E-09	1.22E-09	6E-10	0.967742	1
22	2	3.243243			5.727273			4.1E-08	3.83E-08	3.84E-08	3.83E-08	3.7E-08	0.810811	0.863636
23	1	4	0.341311	4.9228	5.52381	0.341311	4.9228	1.71E-08	3.68E-09	3.53E-09	2.67E-09	1.25E-09	1	1
24	2	3.636364	0.346769	4.591375	5.727273	0.346769	4.591375	1.34E-08	2.83E-09	2.7E-09	2.04E-09	9.47E-10	0.909091	0.909091

25	1	4	0.296721	4.2196	5.52381	0.296721	4.2196	6.97E-09	4.73E-10	4.34E-10	3.3E-10	1.75E-10	1	0.952381
26	2	4	0.311538	4.355216	5.25	0.311538	4.355216	9.52E-09	8.99E-10	8.21E-10	6.42E-10	3.44E-10	1	1
27	3	3.636364	0.273699	2.052331	5.52381	0.273699	2.052331	3.42E-09	5.19E-10	4.99E-10	3.74E-10	3.04E-10	0.909091	1
28	2	3.75	0.389231	17.705	5.478261	0.389231	17.705	2.77E-08	1.88E-08	1.87E-08	1.87E-08	1.84E-08	0.9375	1
29	1	3.870968	0.376	2.882061	5.272727	0.376	2.882061	2.08E-08	4.5E-09	4.3E-09	3.25E-09	1.52E-09	0.967742	1
30	3	4.137931	0.275273	4.63837	5.8	0.275273	4.63837	1.47E-08	1.28E-09	3.97E-10	9.89E-11	2.7E-11	0.827586	0.95
31	2	3.333333	0.252558	6.851524	5.083333	0.252558	6.851524	8.69E-09	4.33E-10	3.37E-10	2.6E-10	1.69E-10	0.944444	0.916667
32	3	3.870968	0.362963	5.558264	5.272727	0.362963	5.558264	9.38E-09	1.79E-09	1.71E-09	1.3E-09	6.39E-10	0.967742	1
33	2	3.428571	0.313818	5.294667	5.083333	0.313818	5.294667	9.12E-09	1.41E-09	1.34E-09	1.02E-09	5.01E-10	0.971429	0.958333
34	3	3.870968	0.404941	3.507429	5.52381	0.404941	3.507429	1.74E-08	3.58E-09	3.43E-09	2.59E-09	1.2E-09	0.967742	1
35	1	3.333333	0.391538	3.099686	5.478261	0.391538	3.099686	5.85E-09	4.94E-10	4.51E-10	3.41E-10	1.85E-10	0.944444	0.956522
36	2	3.428571	0.324727	5.201481	4.88	0.324727	5.201481	1.06E-08	1.29E-09	1.23E-09	9.41E-10	4.84E-10	0.971429	1
37	3	4.615385	0.398413	4.657355	5.52381	0.398413	4.657355	2.26E-08	4.52E-09	4.29E-09	3.23E-09	1.51E-09	0.807692	0.904762
38	2	3.428571			5.083333			2.07E-08	3.71E-09	3.53E-09	2.66E-09	1.24E-09	0.914286	0.916667
39	3	3.428571	0.399178	3.840444	5.52381	0.399178	3.840444	2.57E-08	5.6E-09	5.33E-09	4.01E-09	1.83E-09	0.971429	0.952381
40	4	3.529412	0.366667	4.5056	5.304348	0.366667	4.5056	2.6E-09	5.17E-10	4.9E-10	3.69E-10	2.19E-10	1	0.913043
41	3	3.636364	0.371429	6.835512	5.304348	0.371429	6.835512	1.38E-09	1.78E-10	1.62E-10	1.19E-10	5.22E-11	0.939394	0.913043
42	4	3.529412	0.345217	4.231	5.809524	0.345217	4.231	3.08E-09	5.7E-10	5.29E-10	4.02E-10	1.9E-10	1	0.904762
43	3	3.636364	0.483396	5.663769	5.304348	0.483396	5.663769	1.16E-09	9.7E-11	4.31E-11	3.28E-11	1.46E-11	1	0.956522
44	5	3.428571	0.3568	5.95151	5.809524	0.3568	5.95151	9E-10	1.3E-10	1.05E-10	7.98E-11	4.06E-11	0.885714	0.857143
45	4	3.333333	0.295909	3.25692	5.809524	0.295909	3.25692	1.83E-09	2.69E-10	2.13E-10	1.59E-10	7.89E-11	0.944444	0.857143
46	3	3.870968	0.3775	6.24	5.545455	0.3775	6.24	1.8E-08	2E-08	2E-08	2E-08	1.93E-08	0.870968	0.909091
47	4	4	0.334019	2.786226	5.545455	0.334019	2.786226	2.05E-09	1.05E-09	1.06E-10	7.16E-11	3.11E-11	0.933333	0.909091
48	3	3.529412	0.388276	4.987228	5.083333	0.388276	4.987228	4.57E-09	8.83E-10	8.37E-10	6.24E-10	2.72E-10	0.941176	0.958333
49	5	3.243243	0.412281	5.013714	5.083333	0.412281	5.013714	9.73E-10	8.21E-11	2.92E-11	2.32E-11	1.65E-11	0.891892	1
50	5	3.428571	0.466667	6.778537	5.809524	0.466667	6.778537	9.07E-10	1.25E-10	1.07E-10	8.07E-11	3.56E-11	0.971429	0.904762
51	4	3.529412	0.302951	2.456033	5.304348	0.302951	2.456033	1.36E-09	2.27E-10	1.87E-10	1.35E-10	6.64E-11	1	0.956522
52	3	3.529412	0.396393	4.215533	5.083333	0.396393	4.215533	5.32E-09	8.19E-10	7.75E-10	5.75E-10	2.37E-10	1	0.916667
53	4	3.529412	0.301368	2.57231	5.083333	0.301368	2.57231	1.78E-09	3.16E-10	2.67E-10	1.99E-10	9.94E-11	1	0.958333
54	3	3.529412	0.378974	2.546793	5.304348	0.378974	2.546793	7.31E-10	6.18E-11	2.01E-11	1.5E-11	8.22E-12	1	0.913043
55	4	3.529412	0.302857	4.032348	5.083333	0.302857	4.032348	2.22E-09	1.82E-10	1.44E-10	1.11E-10	6.88E-11	1	0.958333
56	3	3.636364	0.361235	3.66425	4.692308	0.361235	3.66425	2.25E-09	4.22E-10	3.94E-10	2.94E-10	1.32E-10	1	0.884615
57	4	3.529412	0.287789	3.172638	5.304348	0.287789	3.172638	4.41E-09	8.28E-10	7.58E-10	5.71E-10	2.57E-10	1	0.956522

58	3	3.529412	0.335	5.300085	5.083333	0.335	5.300085	3.64E-09	3.34E-10	2.94E-10	2.14E-10	8.27E-11	0.970588	1
59	5	3.529412	0.334074	5.210415	5.545455	0.334074	5.210415	5.96E-09	2.54E-10	1.16E-10	8E-11	3.07E-11	1	0.954545
60	4	4.137931	0.29513	2.484281	5.545455	0.29513	2.484281	1.2E-09	1.56E-10	1.21E-10	8.96E-11	3.75E-11	0.965517	0.954545
61	3	3.428571	0.41942	4.341647	4.88	0.41942	4.341647	1.3E-09	1.85E-10	1.3E-10	9.53E-11	4.39E-11	0.885714	0.92
62	3	3.636364	0.411111	4.810065	4.88	0.411111	4.810065	1.13E-09	1.75E-10	1.08E-10	7.97E-11	3.74E-11	0.909091	1
63	4	3.333333	0.301818	1.728209	5.083333	0.301818	1.728209	1.79E-09	2.4E-10	2.16E-10	1.62E-10	6.91E-11	0.944444	1
64	3	3.333333	0.390345	5.129544	5.384615	0.390345	5.129544	3.71E-09	7.27E-10	6.91E-10	5.15E-10	2.28E-10	1	0.923077
65	4	3	0.288844	2.012192	6.086957	0.288844	2.012192	8.1E-06	6.59E-08	4.22E-08	1.04E-08	1.52E-09	0.975	0.913043
66	5	3.076923	0.365143	4.267304	6.086957	0.365143	4.267304	2.88E-09	5.63E-10	5.18E-10	3.91E-10	1.77E-10	1	0.913043
67	4	3.076923	0.312	2.124259	5.6	0.312	2.124259	4.95E-10	9.1E-11	6.6E-11	4.64E-11	2.16E-11	0.897436	0.92
68	3	3.870968	0.347857	5.300073	5.384615	0.347857	5.300073	2.23E-09	1.8E-10	1.37E-10	1.05E-10	5.78E-11	1	1
69	5	3.243243	0.338286	4.164116	5.6	0.338286	4.164116	1.77E-09	2.75E-10	2.2E-10	1.68E-10	8.34E-11	0.972973	0.96
70	4	3.529412	0.285	2.105874	5.384615	0.285	2.105874	6.81E-10	6.62E-11	4.27E-11	3.02E-11	1.68E-11	1	0.961538
71	3	3.636364	0.389067	3.999243	5.384615	0.389067	3.999243	3.66E-09	7.41E-10	7.01E-10	5.26E-10	2.35E-10	0.939394	0.923077
72	3	3.243243	0.348727	5.518741	5.6	0.348727	5.518741	1.39E-09	1.81E-10	1.43E-10	1.09E-10	5.74E-11	1	0.96
73	4	3.076923	0.294465	1.850987	5.384615	0.294465	1.850987	7.33E-10	1.51E-10	1.21E-10	8.65E-11	3.9E-11	1	1
74	3	3.243243	0.385143	4.310087	5.6	0.385143	4.310087	1.68E-09	3.06E-10	2.67E-10	2.02E-10	9.69E-11	0.972973	0.84
75	4	3	0.30932	2.655843	5.6	0.30932	2.655843	3.05E-09	5.96E-10	5.17E-10	2.72E-10	1.11E-10	1	0.92
76	5	3.333333	0.302319	4.288353	7	0.302319	4.288353	1.87E-09	1.03E-10	4.16E-11	3.15E-11	1.93E-11	0.972222	0.85
77	4	3.243243	0.252289	1.798545	5.384615	0.252289	1.798545	2.1E-09	3.13E-10	2.66E-10	1.97E-10	9.65E-11	0.972973	1
78	3	3.157895	0.363529	3.536571	5.833333	0.363529	3.536571	1.29E-09	2.05E-10	1.77E-10	1.35E-10	6.65E-11	0.894737	1
79	5	3.076923	0.368169	4.180629	5.833333	0.368169	4.180629	1.45E-09	2.62E-10	2.13E-10	1.6E-10	7.27E-11	1	0.958333
80	4	2.926829	0.27675	1.864755	5.384615	0.27675	1.864755	4.8E-10	8.84E-11	6.27E-11	4.19E-11	1.95E-11	0.97561	0.923077
81	3	2.857143	0.455652	6.296444	5.6	0.455652	6.296444	2.56E-09	4.75E-10	4.3E-10	3.25E-10	1.46E-10	0.952381	0.96
82	3	3.243243	0.407143	5.304	5.6	0.407143	5.304	1.8E-09	3.58E-10	3.15E-10	2.38E-10	1.14E-10	1	0.96
83	4	2.926829	0.280714	1.31078	5.384615	0.280714	1.31078	5.06E-10	9.43E-11	6.98E-11	4.91E-11	2.31E-11	0.95122	0.961538
84	5	2.926829	0.322105	3.943627	6.086957	0.322105	3.943627	1.04E-09	1.54E-10	1.14E-10	8.64E-11	4.24E-11	0.97561	0.913043
85	4	3.529412	0.298491	1.856253	6.363636	0.298491	1.856253	3.66E-10	6.45E-11	3.9E-11	2.38E-11	1.05E-11	0.941176	0.909091
86	3	3.076923	0.397143	4.69671	5.384615	0.397143	4.69671	5.23E-09	1.02E-09	9.63E-10	7.2E-10	3.15E-10	0.974359	0.961538
87	5	2.926829	0.423571	5.404436	5.6	0.423571	5.404436	2.93E-09	4.98E-10	4.41E-10	3.37E-10	1.49E-10	0.926829	0.96
88	4	3	0.320822	3.856056	5.185185	0.320822	3.856056	3.98E-10	5.1E-11	2.59E-11	1.45E-11	7.48E-12	1	1
89	3	3	0.40988	3.347805	5.6	0.40988	3.347805	2.82E-09	4.49E-10	3.9E-10	2.91E-10	1.32E-10	1	1
90	5	2.926829	0.354154	4.5755	5.6	0.354154	4.5755	1.71E-09	1.94E-10	9.15E-11	6.86E-11	3.64E-11	0.926829	1

91	4	2.926829	0.312099	1.810236	5.384615	0.312099	1.810236	6.2E-10	1.16E-10	8.81E-11	6.18E-11	2.74E-11	0.97561	1
92	3	2.926829	0.444	3.568811	6.086957	0.444	3.568811	4.68E-09	7.51E-10	7.03E-10	5.21E-10	2.18E-10	0.97561	0.956522
93	3	3	0.344444	5.467245	5.6	0.344444	5.467245	1.41E-09	2.07E-10	1.71E-10	1.28E-10	6.49E-11	1	0.96
94	4	2.926829	0.309885	1.716948	5.833333	0.309885	1.716948	4.58E-10	6.11E-11	3.47E-11	2.1E-11	1.05E-11	0.97561	0.916667
95	4	3	0.312919	1.8649	5.384615	0.312919	1.8649	2.71E-10	4.66E-11	2.18E-11	1.11E-11	5.01E-12	0.975	0.961538
96	6	3	0.372	3.740911	5.185185	0.372	3.740911	3.28E-09	6.68E-10	6.13E-10	4.63E-10	2.06E-10	0.95	1
97	6	3	0.417143	7.12078	5.384615	0.417143	7.12078	4.02E-09	8.08E-10	7.64E-10	5.72E-10	2.55E-10	1	0.961538
98	4	2.790698	0.300828	1.821278	5.384615	0.300828	1.821278	9.97E-10	9.4E-11	6.11E-11	3.98E-11	1.58E-11	0.930233	1
99	5	3	0.440392	5.86152	5.6	0.440392	5.86152	2.87E-09	5.36E-10	4.73E-10	3.57E-10	1.73E-10	0.925	0.96
100	6	3.157895	0.336164	4.136278	5.833333	0.336164	4.136278	2.79E-09	4.77E-10	4.21E-10	3.23E-10	1.61E-10	0.894737	0.875
101	4	2.857143	0.34448	2.368452	5.833333	0.34448	2.368452	3.58E-09	6.83E-10	6.52E-10	4.9E-10	2.36E-10	0.928571	0.916667
102	5	2.926829	0.41625	4.656571	5.384615	0.41625	4.656571	5.25E-09	1.18E-09	1.1E-09	8.36E-10	3.8E-10	0.926829	1
103	6	3	0.388767	4.096889	5.185185	0.388767	4.096889	6.02E-09	1.27E-09	1.21E-09	9.07E-10	4.12E-10	1	0.962963
104	4	3.157895	0.298667	2.211224	5.185185	0.298667	2.211224	4.81E-09	1.02E-09	9.64E-10	7.35E-10	3.49E-10	0.868421	0.962963
105	4	3	0.317818	2.717945	5.833333	0.317818	2.717945	1.91E-09	3.06E-10	2.75E-10	2.09E-10	1.05E-10	1	0.916667
106	6	3	0.350526	3.892853	5.6	0.350526	3.892853	1.33E-09	2.35E-10	1.82E-10	1.34E-10	5.82E-11	0.975	0.96
107	6	2.727273	0.412903	4.897311	6.086957	0.412903	4.897311	9.43E-10	1.05E-10	7.12E-11	5.11E-11	2.45E-11	0.909091	0.913043
108	4	3	0.302727	2.262504	5.6	0.302727	2.262504	2.18E-09	3.92E-10	3.52E-10	2.66E-10	1.28E-10	0.925	0.96
109	5	2.926829	0.401132	5.617154	5.6	0.401132	5.617154	2E-09	3.89E-10	3.53E-10	2.64E-10	1.17E-10	0.95122	0.96
110	5	3.076923	0.402857	7.198537	5.384615	0.402857	7.198537	2.63E-09	5.07E-10	4.55E-10	3.42E-10	1.55E-10	0.897436	0.961538
111	6	3	0.407	7.464923	5.6	0.407	7.464923	1.52E-09	2.39E-10	1.84E-10	1.38E-10	6.41E-11	1	1
112	4	3	0.317647	2.664	5.6	0.317647	2.664	4.09E-09	8.05E-10	7.76E-10	5.7E-10	2.58E-10	1	0.96
113	6	3	0.417391	6.339289	5.6	0.417391	6.339289	3.85E-09	6.3E-10	5.52E-10	4.19E-10	1.86E-10	0.925	0.96
114	4	3.636364	0.214904	1.909462	5.185185	0.214904	1.909462	2.22E-09	2.76E-10	2.2E-10	1.64E-10	8.69E-11	0.848485	1
115	6	2.857143	0.398947	5.164786	4.827586	0.398947	5.164786	1.41E-09	3.06E-10	2.57E-10	1.95E-10	9.19E-11	0.97619	0.931034
116	7	2.857143	0.302857	2.132173	5	0.302857	2.132173	2.54E-09	4.7E-10	4.21E-10	3.2E-10	1.51E-10	1	0.964286
117	5	2.790698	0.352	5.901306	5.833333	0.352	5.901306	1.99E-09	3.53E-10	3.19E-10	2.41E-10	1.12E-10	1	0.791667
118	6	3.157895	0.86	11.392	5.833333	0.86	11.392	2.87E-09	6.37E-10	7.48E-10	8.08E-10	2.07E-10	0.947368	0.875
119	7	2.727273	0.309514	1.608239	6.086957	0.309514	1.608239	2E-09	4.26E-10	3.87E-10	2.93E-10	1.32E-10	1	0.869565
120	6	2.857143	0.443729	5.059034	5.833333	0.443729	5.059034	2.15E-09	3.92E-10	3.53E-10	2.69E-10	1.29E-10	1	0.791667
121	6	2.926829	0.392308	5.792392	5.6	0.392308	5.792392	2.1E-09	3.95E-10	3.48E-10	2.61E-10	1.16E-10	0.97561	0.88
122	7	3	0.327852	2.068776	5.6	0.327852	2.068776	1.47E-09	2.41E-10	2.06E-10	1.56E-10	6.86E-11	0.975	0.96
123	5	2.608696	0.3848	5.985143	5.384615	0.3848	5.985143	4.73E-09	8.93E-10	8.28E-10	6.16E-10	2.68E-10	0.956522	1

124	6	2.727273	0.352381	7.151317	5.6	0.352381	7.151317	6.45E-09	1.16E-09	1.1E-09	8.17E-10	3.5E-10	0.977273	0.88
125	7	2.857143	0.385143	2.140144	5	0.385143	2.140144	2.48E-09	3.9E-10	3.78E-10	2.81E-10	1.24E-10	0.928571	0.964286
126	6	2.926829	0.392881	5.095241	6.086957	0.392881	5.095241	6.37E-09	1.14E-09	1.1E-09	8.19E-10	3.54E-10	1	0.869565
127	7	2.926829	0.287416	1.638576	5	0.287416	1.638576	6.48E-09	1.37E-09	1.26E-09	9.47E-10	4.24E-10	0.95122	1
128	5	3	0.426792	5.655077	4.666667	0.426792	5.655077	6.06E-09	1.15E-09	1.09E-09	8.14E-10	3.52E-10	0.925	0.933333
129	6	2.790698	0.395833	6.332	5.384615	0.395833	6.332	4.51E-09	8.48E-10	7.9E-10	5.94E-10	2.65E-10	0.953488	0.923077
130	7	2.926829	0.303023	1.676094	5.185185	0.303023	1.676094	5.68E-10	9.95E-11	6.41E-11	4.29E-11	2.08E-11	0.97561	0.962963
131	6	2.727273	0.38	7.7832	5.185185	0.38	7.7832	2.84E-08	2.06E-09	4.26E-10	4.35E-11	1.84E-11	1	0.888889
132	7	2.666667	0.328156	1.67382	5.185185	0.328156	1.67382	9.23E-10	1.88E-10	1.56E-10	1.11E-10	5.03E-11	0.977778	0.962963
133	6	2.926829	0.438947	15.95689	5.6	0.438947	15.95689	2.36E-09	4.61E-10	4.13E-10	3.11E-10	1.49E-10	0.804878	0.88
134	6	2.926829	0.453898	5.031655	6.086957	0.453898	5.031655	2.1E-09	4.37E-10	3.91E-10	2.95E-10	1.38E-10	0.926829	0.826087
135	7	2.608696	0.29956	1.62747	5.6	0.29956	1.62747	4.27E-09	8.17E-10	7.56E-10	5.7E-10	2.63E-10	0.956522	0.92
136	6	2.857143	0.499048	7.108976	5.384615	0.499048	7.108976	2.25E-09	4.33E-10	3.81E-10	2.72E-10	1.25E-10	0.952381	0.923077
137	7	2.666667	0.280206	1.395026	5.384615	0.280206	1.395026	4.54E-10	6.91E-11	4.71E-11	3.42E-11	1.66E-11	0.911111	0.923077
138	5	2.926829	0.364727	5.398148	5.384615	0.364727	5.398148	2.51E-09	1.42E-10	7.54E-11	5.61E-11	3.82E-11	0.97561	0.884615
139	5	2.790698	0.441333	6.516909	5	0.441333	6.516909	2.66E-09	5.38E-10	5.05E-10	3.7E-10	1.64E-10	0.976744	0.928571
140	6	2.553191	0.433585	5.706846	5.185185	0.433585	5.706846	1.66E-09	2.64E-10	2.22E-10	1.67E-10	8.17E-11	0.93617	0.925926
141	7	2.790698	0.327397	2.013655	5.6	0.327397	2.013655	5.1E-09	2.6E-10	2.32E-10	1.71E-10	7.63E-11	0.953488	0.88
142	5	2.857143	0.414667	6.686273	5.185185	0.414667	6.686273	3.28E-09	6.51E-10	5.93E-10	4.48E-10	2.04E-10	0.952381	0.962963
143	5	2.727273	0.425263	7.676973	5.185185	0.425263	7.676973	5.46E-09	1.03E-09	9.57E-10	7.1E-10	3.03E-10	0.954545	0.925926
144	6	2.666667	0.382462	4.612813	5.6	0.382462	4.612813	2.57E-09	4.94E-10	4.57E-10	3.5E-10	1.71E-10	0.911111	0.88
145	7	2.790698	0.373	2.181513	5.384615	0.373	2.181513	7.6E-10	1.16E-10	9.81E-11	7.34E-11	3.23E-11	0.953488	0.923077
146	5	2.926829	0.403385	4.585313	6.086957	0.403385	4.585313	2.63E-09	5.18E-10	4.86E-10	3.67E-10	1.75E-10	0.902439	0.826087
147	5	3.076923	0.432903	4.821574	5.185185	0.432903	4.821574	3.12E-09	6.49E-10	6.12E-10	4.6E-10	2.09E-10	0.897436	0.888889
148	6	2.666667	0.4328	6.003347	5.384615	0.4328	6.003347	2.66E-09	6.36E-10	5.98E-10	4.49E-10	2.07E-10	0.977778	0.961538
149	7	2.926829	0.331676	1.677233	5.833333	0.331676	1.677233	3.42E-09	6.89E-11	2.28E-11	8.81E-12	3.6E-12	0.926829	0.875
150	7	2.727273	0.334465	1.843165	5.185185	0.334465	1.843165	7.8E-09	1.18E-10	1.73E-11	5.85E-12	1.66E-12	1	0.925926
151	6	2.727273	0.407857	5.264364	5	0.407857	5.264364	1.9E-09	1.8E-10	1.02E-10	8.04E-11	4.45E-11	0.977273	0.964286
152	7	2.790698	0.295911	1.333304	5.833333	0.295911	1.333304	3.19E-10	6.05E-11	3.51E-11	2.1E-11	9.3E-12	0.953488	0.875
153	6	2.553191	0.401053	5.141	5.185185	0.401053	5.141	2.78E-09	5.57E-10	5.21E-10	3.93E-10	1.82E-10	0.93617	1
154	5	2.5	0.439535	6.98781	5	0.439535	6.98781	3.13E-09	6.23E-10	5.92E-10	4.44E-10	2.04E-10	0.875	0.928571
155	5	2.790698	0.436667	4.851458	5.833333	0.436667	4.851458	2.92E-09	2.62E-10	1.86E-10	1.41E-10	8E-11	0.906977	0.875
156	7	2.790698	0.306667	1.257047	5.6	0.306667	1.257047	6.92E-10	1.3E-10	1.03E-10	7.67E-11	3.46E-11	0.930233	0.92

157	6	2.727273	0.410435	6.454578	5.185185	0.410435	6.454578	2.77E-09	3.33E-10	3.2E-10	2.41E-10	1.29E-10	1	0.925926
158	5	2.790698	0.435484	4.737574	5.185185	0.435484	4.737574	2.83E-09	4.32E-10	3.75E-10	2.73E-10	1.32E-10	1	1
159	7	2.608696	0.34071	1.766571	5.384615	0.34071	1.766571	5.06E-10	1.51E-10	1.28E-10	1.16E-10	1.08E-10	0.956522	0.923077
160	6	2.666667	0.450698	6.987524	6.363636	0.450698	6.987524	3.13E-09	5.7E-10	5.16E-10	3.99E-10	1.81E-10	0.955556	0.818182
161	5	2.790698	0.418974	7.663053	5.185185	0.418974	7.663053	1.75E-09	2.22E-10	1.89E-10	1.43E-10	7.16E-11	0.930233	0.888889
162	7	2.727273	0.312	1.394737	5.185185	0.312	1.394737	3.17E-10	5.66E-11	3.14E-11	1.99E-11	9.43E-12	1	0.962963
163	6	2.553191	0.40303	8.591875	5.6	0.40303	8.591875	4.3E-09	7.95E-10	7.58E-10	5.71E-10	2.54E-10	0.914894	0.88
164	6	2.727273	0.399459	7.975222	5.384615	0.399459	7.975222	3.21E-09	6.51E-10	5.97E-10	4.46E-10	1.98E-10	1	0.884615
165	7	2.666667	0.338012	1.7235	5.384615	0.338012	1.7235	3.85E-09	7.62E-10	7.32E-10	5.45E-10	2.42E-10	0.977778	0.923077
166	7	2.666667	0.277864	1.450615	5.185185	0.277864	1.450615	1.82E-09	3.46E-10	3.15E-10	2.37E-10	1.12E-10	0.933333	0.962963
167	6	2.727273	0.441429	5.099055	5.384615	0.441429	5.099055	4.57E-09	7.91E-10	7.28E-10	5.43E-10	2.45E-10	1	0.923077
168	7	2.608696	0.314419	1.736023	5	0.314419	1.736023	3.76E-10	6.93E-11	4.36E-11	2.76E-11	1.27E-11	0.956522	1
169	8	2.790698	0.42	6.097417	5.384615	0.42	6.097417	2.06E-09	3.43E-10	2.77E-10	2.1E-10	1.01E-10	0.906977	0.961538
170	5	2.608696	0.446939	6.151583	5.6	0.446939	6.151583	2.17E-09	3.7E-10	3.15E-10	2.32E-10	1.06E-10	0.913043	0.96
171	7	2.666667	0.358579	1.637451	5.185185	0.358579	1.637451	4.54E-10	6.74E-11	3.98E-11	2.46E-11	1.19E-11	0.955556	0.925926
172	8	2.44898	0.42	5.579922	5.185185	0.42	5.579922	2.23E-09	3.63E-10	3.09E-10	2.2E-10	1.06E-10	0.979592	0.962963
173	5	2.44898	0.376571	8.451647	5	0.376571	8.451647	4.89E-09	9.25E-10	8.56E-10	6.45E-10	2.8E-10	0.979592	1
174	7	2.4	0.285789	1.315507	5.185185	0.285789	1.315507	3.74E-10	6.37E-11	3.81E-11	2.38E-11	1.16E-11	0.96	1
175	8	2.44898	0.401905	7.050829	5.185185	0.401905	7.050829	4.46E-09	9.17E-10	8.47E-10	6.43E-10	2.9E-10	0.959184	0.962963
176	8	2.5	0.399016	4.948333	5.185185	0.399016	4.948333	3.06E-09	6.16E-10	5.59E-10	4.33E-10	1.98E-10	1	0.888889
177	7	2.4	0.274433	1.50228	5	0.274433	1.50228	7.17E-10	1.54E-10	1.34E-10	1.02E-10	4.53E-11	0.94	1
178	7	2.352941	0.298409	1.701577	5	0.298409	1.701577	2.39E-09	3.92E-10	3.51E-10	2.68E-10	1.32E-10	0.941176	1
179	8	2.44898	0.396667	6.309021	5.185185	0.396667	6.309021	5.21E-09	8.94E-10	8E-10	6.06E-10	2.72E-10	0.979592	1
180	9	2.352941	0.4425	6.274553	5.6	0.4425	6.274553	6.87E-09	1.47E-09	1.39E-09	1.05E-09	4.77E-10	0.941176	0.88
181	8	2.727273	0.370638	6.383217	5.384615	0.370638	6.383217	7.83E-09	1.73E-09	1.64E-09	1.23E-09	5.59E-10	0.977273	0.961538
182	7	2.44898	0.310303	1.508264	5.384615	0.310303	1.508264	1.11E-09	2.04E-10	1.96E-10	1.46E-10	6.74E-11	0.979592	0.961538
183	9	2.44898	0.356667	6.024426	5	0.356667	6.024426	4.26E-09	8.97E-10	8.38E-10	6.33E-10	3.03E-10	0.979592	1
184	7	2.727273	0.295434	1.370661	5.185185	0.295434	1.370661	3.28E-10	5.2E-11	2.74E-11	1.52E-11	7.29E-12	0.886364	0.925926
185	8	2.44898	0.424	7.377128	5	0.424	7.377128	4.76E-09	9.56E-10	9.09E-10	7E-10	3.33E-10	0.959184	0.964286
186	8	2.5	0.348	8.572824	5	0.348	8.572824	2.52E-09	4.84E-10	4.58E-10	3.47E-10	1.65E-10	1	1
187	7	2.4	0.284762	1.417665	4.827586	0.284762	1.417665	3.87E-10	5.22E-11	2.75E-11	1.55E-11	7.45E-12	0.96	1
188	9	2.44898	0.422791	7.016286	5.185185	0.422791	7.016286	5.51E-09	1.2E-09	1.14E-09	8.53E-10	3.88E-10	0.938776	0.851852
189	9	2.5	0.342222	8.444686	5	0.342222	8.444686	5.41E-10	9.45E-11	6.72E-11	4.55E-11	2.02E-11	1	1

190	7	2.44898	0.267547	1.397592	5.384615	0.267547	1.397592	2.15E-09	4.05E-10	3.62E-10	2.77E-10	1.33E-10	0.979592	0.923077
191	8	2.5	0.350476	7.268976	5	0.350476	7.268976	8.1E-09	1.59E-09	1.5E-09	1.12E-09	4.87E-10	1	1
192	8	2.4	0.421212	8.493625	5.185185	0.421212	8.493625	1.96E-09	2.63E-10	2.21E-10	1.71E-10	8.68E-11	0.9	0.962963
193	7	2.5	0.283193	1.241992	5	0.283193	1.241992	5.48E-10	1.06E-10	7.63E-11	5.33E-11	2.58E-11	1	1
194	9	2.4	0.439091	6.938605	5	0.439091	6.938605	1.28E-09	1.92E-10	1.09E-10	8.02E-11	4.02E-11	0.9	1
195	10	2.5	0.323575	1.419961	5.6	0.323575	1.419961	7.43E-10	1.25E-10	9.85E-11	6.85E-11	2.84E-11	1	0.96
196	8	2.264151	0.4216	5.988	5.384615	0.4216	5.988	1.33E-09	1.89E-10	1.53E-10	1.15E-10	5.62E-11	0.886792	0.961538
197	9	2.5	0.295455	6.61386	5.185185	0.295455	6.61386	1.56E-09	2.68E-10	2.08E-10	1.55E-10	7.3E-11	0.916667	0.925926
198	9	2.352941	0.361481	5.518566	5	0.361481	5.518566	1.26E-09	1.45E-10	9.86E-11	7.39E-11	3.66E-11	0.941176	0.964286
199	10	2.553191	0.316768	1.512914	5	0.316768	1.512914	3.23E-10	6.05E-11	3.53E-11	2.13E-11	9.29E-12	0.957447	1
200	8	2.4	0.4	7.198927	5	0.4	7.198927	1.69E-09	3.21E-10	2.82E-10	2.14E-10	9.8E-11	0.96	1
201	10	2.5	0.293455	5.14837	5.6	0.293455	5.14837	2.45E-09	4.43E-10	4.02E-10	3.06E-10	1.46E-10	1	0.88
202	8	2.4	0.44386	5.253786	5.384615	0.44386	5.253786	1.87E-09	2.4E-10	1.34E-10	1.02E-10	5.24E-11	0.94	0.923077
203	8	2.44898	0.402564	7.594	5.6	0.402564	7.594	2.51E-09	5.07E-10	4.74E-10	3.55E-10	1.6E-10	0.979592	0.92
204	10	2.4	0.31075	1.808151	5.185185	0.31075	1.808151	2.62E-10	4.47E-11	1.98E-11	9.5E-12	4.32E-12	0.96	0.962963
205	9	2.44898	0.423846	5.827373	5	0.423846	5.827373	4.41E-09	8.2E-10	7.55E-10	5.62E-10	2.45E-10	0.979592	1
206	10	2.44898	0.397959	6.086167	5	0.397959	6.086167	1.42E-09	2.1E-10	1.72E-10	1.32E-10	6.53E-11	0.979592	0.964286
207	8	2.553191	0.431915	6.141217	5.185185	0.431915	6.141217	2.63E-09	4E-10	3.58E-10	2.72E-10	1.35E-10	0.851064	1
208	9	2.4	0.406667	6.745364	5.833333	0.406667	6.745364	2.09E-09	3.6E-10	3.1E-10	2.28E-10	1.05E-10	0.96	0.875
209	8	2.44898	0.404444	5.601736	5.384615	0.404444	5.601736	2.34E-09	3.87E-10	3.11E-10	2.33E-10	1.02E-10	0.979592	0.961538
210	10	2.4	0.543704	5.506113	5.6	0.543704	5.506113	3.38E-09	3.97E-10	3.56E-10	2.67E-10	1.25E-10	0.96	0.88
211	10	2.608696	0.488235	8.777091	5.833333	0.488235	8.777091	4.06E-09	7.87E-10	7.39E-10	5.58E-10	2.58E-10	0.978261	0.916667
212	8	2.5	0.409091	4.486092	5.185185	0.409091	4.486092	4.41E-09	8.04E-10	7.61E-10	5.66E-10	2.42E-10	1	0.925926
213	8	2.553191	0.457838	8.173889	5.185185	0.457838	8.173889	4.95E-09	9.89E-10	9.29E-10	7.03E-10	3.18E-10	0.957447	0.962963
214	10	2.4			5.833333			3.09E-09	3.96E-10	7.08E-11	1E-11	2.54E-12	0.96	0.916667
215	9	2.44898	0.466349	4.715742	5.185185	0.466349	4.715742	2.82E-09	4.53E-10	3.94E-10	3.02E-10	1.48E-10	0.979592	0.888889
216	8	2.666667	0.406154	5.785882	5.185185	0.406154	5.785882	5.45E-09	9.75E-10	9.07E-10	6.82E-10	3.02E-10	0.933333	0.962963
217	10	2.44898	0.313134	1.45452	4.827586	0.313134	1.45452	3.17E-10	3.76E-11	1.19E-11	3.46E-12	1.29E-12	0.979592	1
218	9	2.4	0.455349	6.879143	5.384615	0.455349	6.879143	2.66E-09	5.08E-10	4.67E-10	3.56E-10	1.7E-10	0.96	0.884615
219	9	2.4	0.367556	6.710091	5	0.367556	6.710091	9.6E-10	9.82E-11	5.19E-11	3.97E-11	2.22E-11	0.94	1
220	10	2.44898	0.30319	1.802864	5.384615	0.30319	1.802864	1.55E-09	2.44E-10	2.06E-10	1.51E-10	7.1E-11	0.979592	0.961538
221	8	2.44898	0.35	6.185362	5.185185	0.35	6.185362	1.31E-09	2.4E-10	2.1E-10	1.57E-10	7.87E-11	0.979592	0.962963
222	9	2.5	0.455897	7.640842	5.185185	0.455897	7.640842	1.38E-09	1.87E-10	1.32E-10	1E-10	4.9E-11	1	0.962963

223	10	2.44898	0.322469	1.840099	5.384615	0.322469	1.840099	8.08E-10	1.41E-10	1.11E-10	7.9E-11	3.31E-11	0.979592	0.961538
224	8	2.352941	0.449787	6.398609	5.833333	0.449787	6.398609	2.65E-09	5.18E-10	4.82E-10	3.65E-10	1.68E-10	0.941176	0.833333
225	9	2.5	0.432923	4.62375	5.185185	0.432923	4.62375	1.94E-09	2.79E-10	2.15E-10	1.61E-10	7.96E-11	1	1
226	9	2.44898	0.410769	5.766039	5	0.410769	5.766039	1.66E-09	3.44E-10	3.11E-10	2.32E-10	1.08E-10	0.959184	1
227	10	2.4	0.322824	1.762627	5.185185	0.322824	1.762627	3.55E-10	6.31E-11	4.23E-11	3.13E-11	1.47E-11	0.96	0.962963
228	8	2.5	0.484348	6.5336	5	0.484348	6.5336	5.03E-09	8.99E-10	8.65E-10	6.49E-10	2.94E-10	1	0.964286
229	9	2.4	0.415294	5.77	4.827586	0.415294	5.77	1.85E-09	1.53E-10	6.18E-11	4.67E-11	2.35E-11	1	1
230	10	2.44898	0.29435	1.68	5	0.29435	1.68	8.63E-10	2.24E-10	5.44E-11	2.85E-11	1.33E-11	0.979592	1
231	9	2.222222	0.414607	3.349909	5.185185	0.414607	3.349909	4.05E-09	8.25E-10	7.78E-10	5.8E-10	2.71E-10	0.944444	0.962963
232	9	2.222222	0.345	6.25234	5	0.345	6.25234	2.13E-09	3.75E-10	3.27E-10	2.52E-10	1.22E-10	0.944444	0.964286
233	9	2.222222	0.418462	5.859216	5.384615	0.418462	5.859216	5.79E-09	1.13E-09	1.03E-09	7.7E-10	3.5E-10	0.944444	0.923077
234	10	2.307692	0.318803	2.562931	5	0.318803	2.562931	5.17E-09	1.04E-09	9.38E-10	7.03E-10	3.17E-10	0.884615	0.964286
235	8	2.44898	0.435714	5.368218	5	0.435714	5.368218	4.4E-09	8.68E-10	8.2E-10	6.12E-10	2.77E-10	0.938776	0.928571
236	9	2.068966	0.432	5.981143	5.384615	0.432	5.981143	4.45E-09	8.94E-10	8.36E-10	6.32E-10	2.96E-10	0.862069	0.884615

Appendix 8: MATLAB code

Clearing the Command Window and Workspace

```
clc

clear
```

Import m-data from spreadsheet

```
opts = spreadsheetImportOptions("NumVariables", 11);



% Specify sheet and range
opts.Sheet = "RA2m";

opts.DataRange = "A2:K237";



% Specify column names and types
opts.VariableNames = ["S", "reactionTimeForEachAttemptInMemoryGame",
"Avg_BlinkDurationInSeconds", "Avg_BlnkInterval", "Avg_Delta",
"Avg_Theta", "Avg_Alpha", "Avg_Beta", "Avg_Gamma", "x_theta_alpha__beta",
"MemoryAccuracy"];

opts.VariableTypes = ["double", "double", "double", "double", "double",
"double", "double", "double", "double", "double", "double"];



% Import the data
mData = readtable("/MATLAB Drive/Lumosity Data.xlsx", opts, "UseExcel",
false);



%% Clear temporary variables
clear opts



% Labeling the Memory Accuracy values to 1 & 0
mAccuracy=rand(size(mData.MemoryAccuracy));

tokeep=mData.MemoryAccuracy<1;

m_0_count=sum(tokeep)

mAccuracy(tokeep)=0;

tokeep=mData.MemoryAccuracy==1;

m_1_count=sum(tokeep)

mAccuracy(tokeep)=1;
```

```
mData = removevars(mData,'MemoryAccuracy');

mData = addvars(mData,mAccuracy);



clear tokeep
```

Import a-data from spreadsheet

```
opts = spreadsheetImportOptions("NumVariables", 11);



% Specify sheet and range
opts.Sheet = "RA2a";

opts.DataRange = "A2:K237";



% Specify column names and types
opts.VariableNames = ["S", "reactionTimeForEachAttemptInAttentionGame",
"Avg_BlinkDurationInSeconds", "Avg_BlnkInterval", "Avg_Delta",
"Avg_Theta", "Avg_Alpha", "Avg_Beta", "Avg_Gamma", "x_theta_alpha__beta",
"AttentionAccuracy"];

opts.VariableTypes = ["double", "double", "double", "double", "double",
"double", "double", "double", "double", "double", "double"];



% Import the data
aData = readtable("/MATLAB Drive/Lumosity Data.xlsx", opts, "UseExcel",
false);



%% Clear temporary variables
clear opts



% Labeling the Attention Accuracy values to 1 & 0
aAccuracy=rand(size(aData.AttentionAccuracy));

tokeep=aData.AttentionAccuracy<1;

a_0_count=sum(tokeep)

aAccuracy(tokeep)=0;

tokeep=aData.AttentionAccuracy==1;

a_1_count=sum(tokeep)

aAccuracy(tokeep)=1;

aData = removevars(aData,'AttentionAccuracy');
```

```
aData = addvars(aData,aAccuracy);


clear tokeep
```

Data Pre-Processing

```
% Fill missing values
aData = fillmissing(aData,"movmean",5);
mData = fillmissing(mData,"movmean",5);
% Fill outliers by cliping them to quartile thresholds
aData = filloutliers(aData,"clip","quartiles");
mData = filloutliers(mData,"clip","quartiles");
% Normalize Data, I have some doubt on this...
aData = normalize(aData,"range");
mData = normalize(mData,"range");
```

Classification Model Construction

k-Nearest Neighbor

```
%%% k-Nearest Neighbor accuracy model
TP = 0;
TN = 0;
FN = 0;
FP = 0;
scores = []; % Store predicted scores
labels = []; % Store true labels


for i = 1:236
    dataTest = aData(i,:);
    dataTrain = aData([1:i-1 i+1:end],:);


    % k-Nearest Neighbor accuracy model
    mdl = fitcknn(dataTrain, "aAccuracy", "NumNeighbors", 4, "Distance",
"jaccard");
```

```
    [preds, scr] = predict(mdl, dataTest);


    labels = [labels; dataTest.aAccuracy];
    scores = [scores; scr(2)]; % Assuming the positive class is 1


    if dataTest.aAccuracy == preds
        if dataTest.aAccuracy == 0
            TP = TP + 1;
        else
            TN = TN + 1;
        end
    end


    if dataTest.aAccuracy ~= preds
        if dataTest.aAccuracy == 0
            FN = FN + 1;
        else
            FP = FP + 1;
        end
    end
end
% ConfusionMatrix, accuracy of the model
confusion_matrix = [TP, FN; FP, TN];
accuracy = (TP + TN) / (TP+FN+FP+TN);
% Sensitivity, Specificity, Precision, F1 Score for aAccuracy
sensitivity = TP / (TP + FN);
specificity = TN / (FP + TN);
precision = TP / (TP + FP);
f1_score = 2 * (precision * sensitivity) / (precision + sensitivity);


% Plot ROC curve
 [X, Y, ~, AUC] = perfcurve(labels, scores, 1);
plot(X, Y);
```

```
xlabel('False Positive Rate');

ylabel('True Positive Rate');

title('Receiver Operating Characteristic (ROC) Curve');

legend(['AUC = ', num2str(AUC)]);

hold off;

% Set figure properties

fig = gcf;

fig.PaperUnits = 'inches';

fig.PaperPosition = [0 0 8 6]; % Adjust the size as per your requirement
(8x6 inches in this case)

fig.PaperPositionMode = 'manual';

% Save the plot with 600dpi resolution

print('kNN_roc_plot_attention', '-dpng', '-r600');


% Display results
fprintf('kNN attention model');

fprintf('Confusion Matrix:\n');

disp(confusion_matrix);

fprintf('Accuracy: %.2f%%\n', accuracy * 100);

fprintf('Specificity: %.2f%%\n', specificity * 100);

fprintf('Sensitivity: %.2f%%\n', sensitivity * 100);

fprintf('Precision: %.2f%%\n', precision * 100);

fprintf('F1 Score: %.2f\n', f1_score);

fprintf('Area Under the Curve (AUC): %.2f\n', AUC);




%%% k-Nearest Neighbor memory model
TP = 0;

TN = 0;

FN = 0;

FP = 0;

scores = []; % Store predicted scores

labels = []; % Store true labels
```

```
for i = 1:236
    dataTest = mData(i,:);

    dataTrain = mData([1:i-1 i+1:end],:);


    % k-Nearest Neighbor memory model
    mdl = fitcknn(dataTrain, "mAccuracy", "NumNeighbors", 4, "Distance",
"jaccard");

    [preds, scr] = predict(mdl, dataTest);


    labels = [labels; dataTest.mAccuracy];
    scores = [scores; scr(2)]; % Assuming the positive class is 1


    if dataTest.mAccuracy == preds
        if dataTest.mAccuracy == 0

            TP = TP + 1;

        else

            TN = TN + 1;

        end

    end


    if dataTest.mAccuracy ~= preds
        if dataTest.mAccuracy == 0

            FN = FN + 1;

        else

            FP = FP + 1;

        end

    end

end


% ConfusionMatrix, accuracy of the model
confusion_matrix = [TP, FN; FP, TN];

accuracy = (TP + TN) / (TP+FN+FP+TN);

% Sensitivity, Specificity, Precision, F1 Score for aAccuracy
```

```
sensitivity = TP / (TP + FN);

specificity = TN / (FP + TN);

precision = TP / (TP + FP);

f1_score = 2 * (precision * sensitivity) / (precision + sensitivity);


% Plot ROC curve
 [X, Y, ~, AUC] = perfcurve(labels, scores, 1);

plot(X, Y);

xlabel('False Positive Rate');

ylabel('True Positive Rate');

title('Receiver Operating Characteristic (ROC) Curve');

legend(['AUC = ', num2str(AUC)]);

hold off;

% Set figure properties

fig = gcf;

fig.PaperUnits = 'inches';

fig.PaperPosition = [0 0 8 6]; % Adjust the size as per your requirement
(8x6 inches in this case)

fig.PaperPositionMode = 'manual';

% Save the plot with 600dpi resolution

print('kNN_roc_plot_memory', '-dpng', '-r600');


% Display results
fprintf('kNN memory model');

fprintf('Confusion Matrix:\n');

disp(confusion_matrix);

fprintf('Accuracy: %.2f%%\n', accuracy * 100);

fprintf('Specificity: %.2f%%\n', specificity * 100);

fprintf('Sensitivity: %.2f%%\n', sensitivity * 100);

fprintf('Precision: %.2f%%\n', precision * 100);

fprintf('F1 Score: %.2f\n', f1_score);

fprintf('Area Under the Curve (AUC): %.2f\n', AUC);
```

Decision Tree

```
%%% Decision Tree accuracy model

TP = 0;

TN = 0;

FN = 0;

FP = 0;

bestlevel1=0;

scores = []; % Store predicted scores

labels = []; % Store true labels


for i = 1:236
    dataTest = aData(i,:);

    dataTrain = aData([1:i-1 i+1:end],:);


    % Decision Tree accuracy model
    mdl=fitctree(aData,"aAccuracy");

    [~,~,~,bestlevel] = cvLoss(mdl,...

    'SubTrees','All','TreeSize','min');

    pmdl = prune(mdl,"Level",4);

    [preds, scr] = predict(pmdl, dataTest);


    labels = [labels; dataTest.aAccuracy];
    scores = [scores; scr(2)]; % Assuming the positive class is 1


    if dataTest.aAccuracy == preds
        if dataTest.aAccuracy == 0

            TP = TP + 1;

        else

            TN = TN + 1;

        end

    end


    if dataTest.aAccuracy ~= preds
        if dataTest.aAccuracy == 0
```

```
            FN = FN + 1;
        else
            FP = FP + 1;
        end
    end
    bestlevel1=bestlevel+bestlevel1;
end
bestlevel_to_prune=bestlevel1/(TP+FN+FP+TN)
% ConfusionMatrix, accuracy of the model
confusion_matrix = [TP, FN; FP, TN];
accuracy = (TP + TN) / (TP+FN+FP+TN);
% Sensitivity, Specificity, Precision, F1 Score for aAccuracy
sensitivity = TP / (TP + FN);
specificity = TN / (FP + TN);
precision = TP / (TP + FP);
f1_score = 2 * (precision * sensitivity) / (precision + sensitivity);


% Plot ROC curve
 [X, Y, ~, AUC] = perfcurve(labels, scores, 1);
plot(X, Y);
xlabel('False Positive Rate');
ylabel('True Positive Rate');
title('Receiver Operating Characteristic (ROC) Curve');
legend(['AUC = ', num2str(AUC)]);
hold off;
% Set figure properties
fig = gcf;
fig.PaperUnits = 'inches';
fig.PaperPosition = [0 0 8 6]; % Adjust the size as per your requirement
(8x6 inches in this case)
fig.PaperPositionMode = 'manual';
% Save the plot with 600dpi resolution
print('DT_roc_plot_attention', '-dpng', '-r600');
```

```
% Display results
fprintf('Decision Tree attention model');
fprintf('Confusion Matrix:\n');
disp(confusion_matrix);
fprintf('Accuracy: %.2f%%\n', accuracy * 100);
fprintf('Specificity: %.2f%%\n', specificity * 100);
fprintf('Sensitivity: %.2f%%\n', sensitivity * 100);
fprintf('Precision: %.2f%%\n', precision * 100);
fprintf('F1 Score: %.2f\n', f1_score);
fprintf('Area Under the Curve (AUC): %.2f\n', AUC);



%%% Decision Tree memory model
TP = 0;
TN = 0;
FN = 0;
FP = 0;
bestlevel1=0;
scores = []; % Store predicted scores
labels = []; % Store true labels

for i = 1:236
    dataTest = mData(i,:);
    dataTrain = mData([1:i-1 i+1:end],:);

    % Decision Tree memory model
    mdl=fitctree(mData,"mAccuracy");
    [~,~,~,bestlevel] = cvLoss(mdl,...
    'SubTrees','All','TreeSize','min');
    pmdl = prune(mdl,"Level",3);
    [preds, scr] = predict(pmdl, dataTest);
```

```
    labels = [labels; dataTest.mAccuracy];
    scores = [scores; scr(2)]; % Assuming the positive class is 1


    if dataTest.mAccuracy == preds
        if dataTest.mAccuracy == 0

            TP = TP + 1;

        else

            TN = TN + 1;

        end

    end


    if dataTest.mAccuracy ~= preds
        if dataTest.mAccuracy == 0

            FN = FN + 1;

        else

            FP = FP + 1;

        end

    end

    bestlevel1=bestlevel+bestlevel1;

end

bestlevel_to_prune=bestlevel1/(TP+FN+FP+TN)

% ConfusionMatrix, accuracy of the model

confusion_matrix = [TP, FN; FP, TN];

accuracy = (TP + TN) / (TP+FN+FP+TN);

% Sensitivity, Specificity, Precision, F1 Score for aAccuracy

sensitivity = TP / (TP + FN);

specificity = TN / (FP + TN);

precision = TP / (TP + FP);

f1_score = 2 * (precision * sensitivity) / (precision + sensitivity);


% Plot ROC curve
 [X, Y, ~, AUC] = perfcurve(labels, scores, 1);

plot(X, Y);
```

```
xlabel('False Positive Rate');
ylabel('True Positive Rate');
title('Receiver Operating Characteristic (ROC) Curve');
legend(['AUC = ', num2str(AUC)]);
hold off;
% Set figure properties
fig = gcf;
fig.PaperUnits = 'inches';
fig.PaperPosition = [0 0 8 6]; % Adjust the size as per your requirement
(8x6 inches in this case)
fig.PaperPositionMode = 'manual';
% Save the plot with 600dpi resolution
print('DT_roc_plot_memory', '-dpng', '-r600');


% Display results
fprintf('Decision Tree memory model');
fprintf('Confusion Matrix:\n');
disp(confusion_matrix);
fprintf('Accuracy: %.2f%%\n', accuracy * 100);
fprintf('Specificity: %.2f%%\n', specificity * 100);
fprintf('Sensitivity: %.2f%%\n', sensitivity * 100);
fprintf('Precision: %.2f%%\n', precision * 100);
fprintf('F1 Score: %.2f\n', f1_score);
fprintf('Area Under the Curve (AUC): %.2f\n', AUC);
```

Support Vector Machines

```
%%% Support Vector Machines accuracy model
TP = 0;
TN = 0;
FN = 0;
FP = 0;
scores = []; % Store predicted scores
labels = []; % Store true labels
```

```
for i = 1:236
    dataTest = aData(i,:);

    dataTrain = aData([1:i-1 i+1:end],:);


    % Support Vector Machines accuracy model
    mdl=fitcsvm(aData,"aAccuracy","KernelFunction","polynomial");

    [preds, scr] = predict(mdl, dataTest);


    labels = [labels; dataTest.aAccuracy];
    scores = [scores; scr(2)]; % Assuming the positive class is 1


    if dataTest.aAccuracy == preds
        if dataTest.aAccuracy == 0

            TP = TP + 1;

        else

            TN = TN + 1;

        end

    end


    if dataTest.aAccuracy ~= preds
        if dataTest.aAccuracy == 0

            FN = FN + 1;

        else

            FP = FP + 1;

        end

    end
end

% ConfusionMatrix, accuracy of the model

confusion_matrix = [TP, FN; FP, TN];

accuracy = (TP + TN) / (TP+FN+FP+TN);

% Sensitivity, Specificity, Precision, F1 Score for aAccuracy

sensitivity = TP / (TP + FN);

specificity = TN / (FP + TN);
```

```
precision = TP / (TP + FP);

f1_score = 2 * (precision * sensitivity) / (precision + sensitivity);


% Plot ROC curve
 [X, Y, ~, AUC] = perfcurve(labels, scores, 1);

plot(X, Y);

xlabel('False Positive Rate');

ylabel('True Positive Rate');

title('Receiver Operating Characteristic (ROC) Curve');

legend(['AUC = ', num2str(AUC)]);

hold off;

% Set figure properties

fig = gcf;

fig.PaperUnits = 'inches';

fig.PaperPosition = [0 0 8 6]; % Adjust the size as per your requirement
(8x6 inches in this case)

fig.PaperPositionMode = 'manual';

% Save the plot with 600dpi resolution

print('SVM_roc_plot_attention', '-dpng', '-r600');


% Display results
fprintf('Support Vector Machines attention model');

fprintf('Confusion Matrix:\n');

disp(confusion_matrix);

fprintf('Accuracy: %.2f%%\n', accuracy * 100);

fprintf('Specificity: %.2f%%\n', specificity * 100);

fprintf('Sensitivity: %.2f%%\n', sensitivity * 100);

fprintf('Precision: %.2f%%\n', precision * 100);

fprintf('F1 Score: %.2f\n', f1_score);

fprintf('Area Under the Curve (AUC): %.2f\n', AUC);



%%% Support Vector Machines memory model
```

```
TP = 0;

TN = 0;

FN = 0;

FP = 0;

scores = []; % Store predicted scores

labels = []; % Store true labels


for i = 1:236
    dataTest = mData(i,:);

    dataTrain = mData([1:i-1 i+1:end],:);


    % Support Vector Machines memory model
    mdl=fitcsvm(mData,"mAccuracy","KernelFunction","polynomial");

    [preds, scr] = predict(mdl, dataTest);


    labels = [labels; dataTest.mAccuracy];
    scores = [scores; scr(2)]; % Assuming the positive class is 1


    if dataTest.mAccuracy == preds
        if dataTest.mAccuracy == 0

            TP = TP + 1;

        else

            TN = TN + 1;

        end

    end


    if dataTest.mAccuracy ~= preds
        if dataTest.mAccuracy == 0

            FN = FN + 1;

        else

            FP = FP + 1;

        end

    end

end
```

```
% ConfusionMatrix, accuracy of the model
confusion_matrix = [TP, FN; FP, TN];

accuracy = (TP + TN) / (TP+FN+FP+TN);

% Sensitivity, Specificity, Precision, F1 Score for aAccuracy

sensitivity = TP / (TP + FN);

specificity = TN / (FP + TN);

precision = TP / (TP + FP);

f1_score = 2 * (precision * sensitivity) / (precision + sensitivity);


% Plot ROC curve
 [X, Y, ~, AUC] = perfcurve(labels, scores, 1);

plot(X, Y);

xlabel('False Positive Rate');

ylabel('True Positive Rate');

title('Receiver Operating Characteristic (ROC) Curve');

legend(['AUC = ', num2str(AUC)]);

hold off;

% Set figure properties

fig = gcf;

fig.PaperUnits = 'inches';

fig.PaperPosition = [0 0 8 6]; % Adjust the size as per your requirement
(8x6 inches in this case)

fig.PaperPositionMode = 'manual';

% Save the plot with 600dpi resolution

print('SVM_roc_plot_memory', '-dpng', '-r600');

% Display results

fprintf('Support Vector Machines memory model');

fprintf('Confusion Matrix:\n');

disp(confusion_matrix);

fprintf('Accuracy: %.2f%%\n', accuracy * 100);

fprintf('Specificity: %.2f%%\n', specificity * 100);

fprintf('Sensitivity: %.2f%%\n', sensitivity * 100);
```

```
fprintf('Precision: %.2f%%\n', precision * 100);

fprintf('F1 Score: %.2f\n', f1_score);

fprintf('Area Under the Curve (AUC): %.2f\n', AUC);
```

Neural Network

```
%%% Neural Network accuracy model

TP = 0;

TN = 0;

FN = 0;

FP = 0;

scores = []; % Store predicted scores

labels = []; % Store true labels


for i = 1:236
    dataTest = aData(i,:);

    dataTrain = aData([1:i-1 i+1:end],:);


    % Neural Network accuracy model
    mdl=fitcnet(aData,"aAccuracy","LayerSizes",5);

    [preds, scr] = predict(mdl, dataTest);


    labels = [labels; dataTest.aAccuracy];
    scores = [scores; scr(2)]; % Assuming the positive class is 1


    if dataTest.aAccuracy == preds
        if dataTest.aAccuracy == 0

            TP = TP + 1;

        else

            TN = TN + 1;

        end

    end


    if dataTest.aAccuracy ~= preds
        if dataTest.aAccuracy == 0
```

```
            FN = FN + 1;
        else
            FP = FP + 1;
        end
    end
end
% ConfusionMatrix, accuracy of the model
confusion_matrix = [TP, FN; FP, TN];
accuracy = (TP + TN) / (TP+FN+FP+TN);
% Sensitivity, Specificity, Precision, F1 Score for aAccuracy
sensitivity = TP / (TP + FN);
specificity = TN / (FP + TN);
precision = TP / (TP + FP);
f1_score = 2 * (precision * sensitivity) / (precision + sensitivity);


% Plot ROC curve
 [X, Y, ~, AUC] = perfcurve(labels, scores, 1);
plot(X, Y);
xlabel('False Positive Rate');
ylabel('True Positive Rate');
title('Receiver Operating Characteristic (ROC) Curve');
legend(['AUC = ', num2str(AUC)]);
hold off;
% Set figure properties
fig = gcf;
fig.PaperUnits = 'inches';
fig.PaperPosition = [0 0 8 6]; % Adjust the size as per your requirement
(8x6 inches in this case)
fig.PaperPositionMode = 'manual';
% Save the plot with 600dpi resolution
print('NN_roc_plot_attention', '-dpng', '-r600');


% Display results
```

```
fprintf('Neural Network attention model');

fprintf('Confusion Matrix:\n');

disp(confusion_matrix);

fprintf('Accuracy: %.2f%%\n', accuracy * 100);

fprintf('Specificity: %.2f%%\n', specificity * 100);

fprintf('Sensitivity: %.2f%%\n', sensitivity * 100);

fprintf('Precision: %.2f%%\n', precision * 100);

fprintf('F1 Score: %.2f\n', f1_score);

fprintf('Area Under the Curve (AUC): %.2f\n', AUC);




%%% Neural Network memory model
TP = 0;

TN = 0;

FN = 0;

FP = 0;

scores = []; % Store predicted scores

labels = []; % Store true labels


for i = 1:236
    dataTest = mData(i,:);

    dataTrain = mData([1:i-1 i+1:end],:);


    % Neural Network memory model
    mdl=fitcnet(mData,"mAccuracy","LayerSizes",5);

    [preds, scr] = predict(mdl, dataTest);


    labels = [labels; dataTest.mAccuracy];
    scores = [scores; scr(2)]; % Assuming the positive class is 1


    if dataTest.mAccuracy == preds
        if dataTest.mAccuracy == 0

            TP = TP + 1;
```

```
        else
            TN = TN + 1;
        end
    end

    if dataTest.mAccuracy ~= preds
        if dataTest.mAccuracy == 0
            FN = FN + 1;
        else
            FP = FP + 1;
        end
    end
end

% ConfusionMatrix, accuracy of the model
confusion_matrix = [TP, FN; FP, TN];
accuracy = (TP + TN) / (TP+FN+FP+TN);
% Sensitivity, Specificity, Precision, F1 Score for aAccuracy
sensitivity = TP / (TP + FN);
specificity = TN / (FP + TN);
precision = TP / (TP + FP);
f1_score = 2 * (precision * sensitivity) / (precision + sensitivity);

% Plot ROC curve
 [X, Y, ~, AUC] = perfcurve(labels, scores, 1);
plot(X, Y);
xlabel('False Positive Rate');
ylabel('True Positive Rate');
title('Receiver Operating Characteristic (ROC) Curve');
legend(['AUC = ', num2str(AUC)]);
hold off;
% Set figure properties
fig = gcf;
```

```
fig.PaperUnits = 'inches';

fig.PaperPosition = [0 0 8 6]; % Adjust the size as per your requirement
(8x6 inches in this case)

fig.PaperPositionMode = 'manual';

% Save the plot with 600dpi resolution

print('NN_roc_plot_memory', '-dpng', '-r600');


% Display results
fprintf('Neural Network memory model');

fprintf('Confusion Matrix:\n');

disp(confusion_matrix);

fprintf('Accuracy: %.2f%%\n', accuracy * 100);

fprintf('Specificity: %.2f%%\n', specificity * 100);

fprintf('Sensitivity: %.2f%%\n', sensitivity * 100);

fprintf('Precision: %.2f%%\n', precision * 100);

fprintf('F1 Score: %.2f\n', f1_score);

fprintf('Area Under the Curve (AUC): %.2f\n', AUC);
```